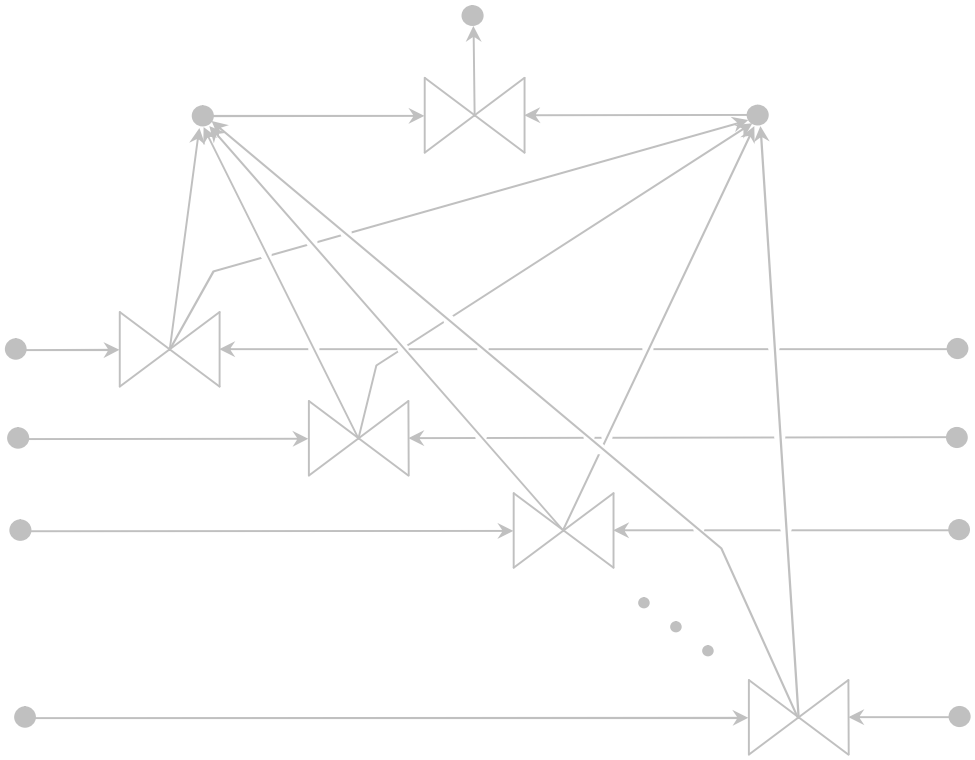
Design Patterns of Successful Role-Playing Games



by
Whitson John Kirk III

Edited by
Michael R. Cantrell

Foreword by
Mike Holmes

9 / 13 / 2009

Table of Contents

Dedication

I dedicate this book to my loving wife Melissa, who has not only patiently endured my obsession with role-playing over the many years of our marriage, but has damned well made sure I did a proper job of it.

Foreword

I first got to know John Kirk through The Forge, and then giving him some design commentary for his game Legendary Quest (www.legendaryquest.com). John, well versed in legend, had put together a lot of research for the game, but the system was pretty traditional in some ways. But with definite potential. Often times when you try to give advice to an author like this, they decide that you're an obnoxious poof, and ignore you completely.

John, on the contrary, took to theory and design ideas like a sponge. A fellow programmer, and educated as an engineer, John understood implicitly that there are simply better and worse ways to approach any process. And that you at least had to know what you were dealing with in detail. So he started really looking around at other game systems to see what the state of the art was, and just how it was that people approached different problems. While this benefited his designs somewhat, after a while John announced to me that what he wanted to do was to write this book. To emulate what had been done in programming in terms of enumerating the methods which people use in RPG designs. He wasn't the first to propose doing this, and given the rate at which his game design tended to advance, I was skeptical that he'd be the one to do it.

But I should have realized, given his background in research and programming, that John was precisely the man for the job. More than that, he'd cracked the essential problem in terms of making such a document, how to partition the information such that it could be presented in a manner that made sense to the reader in terms of how one method is distinct from another. He had a template to work from, and all he had to do was to fill in the blanks. That's a lot of blanks (as you'll see), however.

So he gutted it out, and what's here is the product of that effort. At the time of this writing, the document is in a sort of a "beta" format. He and I have batted it back and forth a bit, but we've realized that it's now time to get more hands on it. That is, it's understood that his research couldn't possibly be entirely comprehensive, and that some reorganization is probably in order. But it's more than just a start, it's got enough meat on it that much of it will stand as written, and those adjustments to it will be informed by what is already there.

I think that John is doing a great service to the community by presenting this book in that, if it is accepted by the community, it will have taken another leap forward in creating a shared vocabulary for us that, started by Ron Edwards et. Al. at The Forge, has served to make it possible to have intelligible discussions about these matters. So take it for what it is, a tremendous effort at organizing the design elements found present in RPGs. Using these definitions and notations about them, and adding to them, I think this will be an important tool for the design community going forward. That is, it will be as good a tool as we all hone it to be.

Mike Holmes

Acknowledgements

Before writing this book, I had been designing, writing, tweaking, and rewriting my own role-playing game, Legendary Quest, for over 20 years. In that time, I created seven editions of the game to which only my close friends and I had access. The experience gave me great practice in designing games and taught me much about how a role-playing game should work. Because I had approached my designs from the vantage point of making the best game possible for my gaming group, I didn't look too far afield for how other games approached similar problems. My friends and I created designs that suited our interests and we were happy with the results. And, in fact, we should have been pleased. Through our efforts, Legendary Quest evolved into a fine game. I would like to thank all the LQ play-testers over the years in helping me in my various game designs. I would especially like to thank Matt Ault, Dave Bailey, James Bockmon, Mike Brown, Denys Carrico-Bockmon, Mark Chester, Leroy Hills, Melissa Kirk, Mike Patrick, Adam Reid, and Paul White for all of their support and many design ideas over the years.

When I made Legendary Quest available on the Internet, everything changed. A fan base sprouted up that began providing feedback. And, I started interacting with other game authors, primarily on The Forge website (http://www.indie-rpgs.com). To my delight, I discovered that I still had a lot to learn about game design. Several years ago, the Forge was a forum devoted to exploring RPG Game Theory and creating well-crafted independent role-playing games. To a game designer, The Forge was candy store, amusement park, and Christmas all wrapped up into one. It kept me enthralled for years, although my role was primarily been that of a silent lurker. Only rarely did I contribute back, and then only when I thought I could provide some insight that others had overlooked. That didn't happen often. Unfortunately, not long after I put up an initial draft of this book on The Forge, the administrators made the decision to de-emphasize the development of RPG Game Theory on their site and focus instead on crafting new games. They locked the forum discussing RPG Theory, so visitors could read the discussions that had taken place earlier, but prevented anyone from making new posts. At that point, I largely lost interest in the site, although I still drop by from time to time.

Even so, I have always wanted to contribute something back to the community. To date, this book is my best attempt at doing that. So, even though they do not know me very well (if at all), I would like to thank the following people on The Forge for the inspiration they have given me over the years: Vincent Baker, Paul Czege, Ron Edwards, John Kim, Timothy Kleinert, Chris Lehrich, Tony Lower-Basch, Ralph Mazza, Clinton R. Nixon, Jared A. Sorensen, and M. J. Young.

There is one other Forge-ite to which I would like to express my deepest gratitude. Mike Holmes took me under his wing early on in my game studies. He has provided me with a tremendous amount of constructive criticism on my designs and has patiently tutored me on modern thoughts in game design theory. I have never encountered anyone with as much sheer volume of knowledge about various role-playing games as Mike. This work would be far less useful without his insight.

Finally, I would also like to give special thanks to Mike Cantrell for hosting and administering the LQ Design Forums and for editing this book. Many people pointed out numerous flaws in an early draft of RPG Design Patterns. So, it was obvious that I was in great need of Mike's technical writing expertise. I believe the book to be greatly enhanced by his efforts.

John Kirk

(9/13/2009)

Introduction

In the 1960s, Christopher Alexander, an architect, proposed a practical new way to undertake urban planning. His idea was to first study the best examples of contemporary urban plans and buildings with the goal of finding common patterns in their designs. Once identified, these patterns could be exploited in future designs. He described this process of design by pattern (or, in his terms, diagrams) in his work Notes on the Synthesis of Form. In this text, Alexander describes patterns as being not merely informal guidelines, but as a formalized arena of discourse. Once a pattern is identified and formalized, it can be easily referenced by domain experts and objectively compared to other formalized patterns in its ability to satisfy design goals.

In 1987, Christopher Alexander's ideas were first applied to software when Ward Cunningham and Kent Beck wrote a paper entitled "Using Pattern Languages for Object-Oriented Programs." This paper presented five patterns that could be used to solve problems in graphical user interface (GUI) design. The software community saw the potential of design patterns and a great deal of discussion ensued in articles and workshops.

Seven years later (1995), the book Design Patterns: Elements of Reusable Object-Oriented Software was published. This book was the first to bring the concept of design patterns to the software development community at large. In so doing, the book revolutionized how modern software is written. The book is so well respected that the four authors who wrote it are known simply as "The Gang of Four" by developers subscribing to the design pattern philosophy. The book consists of 24 design patterns that instruct the reader in excellent solutions to common software problems. The book is considered seminal and its pattern names have become common industry jargon.

The authors of Design Patterns looked at the designs of many, many successful software systems and interviewed their programmers concerning their design decisions. Note that only *successful* programs were investigated, since they weren't trying to analyze why projects fail, but merely to find common characteristics of successful designs. No matter how inventive a solution, though, the authors would not call it a "pattern" unless it clearly appeared in at least two independent systems.

In all of their analyses, a number of patterns emerged. Certain solutions were found again and again. The authors took their results and formally wrote up detailed descriptions of the patterns and the problems they solved. By doing so, they elevated the "rules of thumb" they encountered to fundamental design principles. Once their book was published, even the gurus benefited, since they now had access to a treasure-trove of world-class design solutions, many of which would have been new to even the best of them.

Although my formal training is in Engineering, I am a software developer by trade with many years of experience in architecting software systems. It is probably no surprise, then, that my primary role-playing game design project, Legendary Quest, has been

heavily influenced by software design concepts. I believe that role-playing games in general can profit by the lessons learned by the software industry. Consequently, I firmly believe that role-playing games can benefit from the same kind of design pattern analysis undertaken by the Gang of Four. It seems obvious to me that patterns exist in the design of role-playing games. If we analyze successful games, we should be able to identify Role-Playing Game (RPG) design patterns that could be re-used in future game designs.

Software design patterns do not form the basis for any software theory, although they may exploit theoretical concepts such as object orientation. Similarly, RPG design patterns will not likely form the basis for any new RPG theory, such as Gamism/Narrativism/Simulationism (GNS), Game/Drama/Simulation (GDS), The Big Model, or any other. (If you don't know what any of those terms mean, don't worry, you don't need them to understand this book. If you want to learn more about RPG theory, though, visit The Forge website at http://www.indie-rpgs.com.)

RPG design patterns are not about deciding upon a creative agenda, genre, or even helping you clarify your design goals. RPG design patterns are about formalizing the mechanics observed in existing pen-and-paper role-playing games, the kind of games where the players sit around a table and actually talk to one another. Each pattern description discusses its particular strengths and weaknesses, and educates game designers interested in using the same techniques on how to properly implement them.

This study focuses exclusively on the nitty-gritty structure and mechanical design of table-top role-playing systems. RPG design patterns have nothing to do with mood or setting, although these issues are obviously quite important to many games. In other words, design patterns approach game development at a micro level rather than a macro level. For example, if you have decided that you want to abandon hit points as a means to measure character survivability in your fledgling game, what are your other options? What about alignment? Are there better ways to guide character behavior? Are character classes the best option for your design goals? If so, what pitfalls should you avoid in implementing them? If not, what are the alternatives? How should conflicts be resolved? Is there more than one approach?

RPG design patterns should be kept as independent as possible from over-arching game theories. Design patterns fall into the realm of RPG engineering rather than RPG theory. So, design patterns should be viewed as complementary to theory rather than competitive. Most patterns will be at a very low level. The only assumption RPG design patterns should make is that the designs of good role-playing games re-use common patterns that can be identified and exploited in the designs of future games. Understanding design patterns, then, helps a game designer avoid repeating the same low-level mechanical mistakes others have made in the past. RPG Theory, on the other hand, looks at the whole problem of game design from a high level. It analyzes broader issues such as the differences between games that emphasize story creation over competition, and how these differences impact play.

The First Step in Designing an RPG

Design Patterns won't help you to design anything if you don't have an idea of what it is you are designing. First of all, let's suppose you have the following goals:

1) You want to design a "pen and paper" role-playing game, where real people sit around a real table and talk to one another face to face.
2) You want the game to be fun.

Good. That narrows down the scope of things you might be designing from, say, a hydraulic back hoe to something that this book can help you with. But, while those goals certainly help get us into this particular ballpark, they aren't sufficient for this book to do you any real good. Before you start looking at specific design patterns, you need to have some clear idea of exactly what kind of game you are going about designing. RPG design patterns help you objectively decide what to include in your game based on your design goals. Without those goals, you've got nothing to guide you.

So, before you start, sit down and figure out precisely what it is that your game is going to be about. What are you trying to accomplish? What mood are you trying to evoke? What do the characters *do*? More importantly, what do the *players* do? What literary genre corresponds to your concept? What age group does your game target? What kind of activities do you want to reward and what kinds of rewards do you want to provide? Are you concerned about implementing this game in a computer at any point in the future? Are you expecting your game sequences to extend for many sessions or will stories play out relatively quickly? Do you envision a game that can be easily extended with numerous supplements or are you more interested in creating a single, self-contained book of simple rules? The more specific you are in stating your goals, the better off you'll be and the more useful advice this book can provide.

Definitions

Needless to say, the research for this book entailed reading a lot of role-playing games. But, comprehension did not come easily. Picking up a new game and thumbing through its pages to extract the important bits was often painful. Game authors use popular terms like "attribute," "skill," and "trait" inconsistently. So, understanding an author's intent required digging through the game text and then translating the terms into a common vernacular. Note that some of these terms are written up as full-blown design patterns.

Attribute: A *gauge* that is a common characteristic, a commonality. (See, gauge definition, Attribute design pattern.)

Character: A persona in a game portrayed by a player, including possibly the *Game Master*.

Characteristic: An aspect of a character. A character's name, height, age, beauty, and strength are some possible characteristics.

Common Characteristic (Attribute or **Commonality):** A characteristic common to all characters of a given type in a game. A character's name, height, age, beauty, and strength are frequently common characteristics. Most games directly state the attributes and other commonalities characters possess. Some games provide different sets of attributes for player and non-player characters. Such games partition PC's and NPC's into different types.

Conflict: Contention between characters, players, and/or game forces, especially contention that shapes the game's plot. This includes opposition between two or more players concerning what facts should be introduced into a game world.

Contest: A conflict that is resolved through mechanical means (e.g., dice rolls, comparing numbers, etc.).

Derived Attribute: An attribute whose value is determined by a formula. Typically the formula uses other attribute values to generate a number.

Drama: An outcome based purely on story considerations. A drama based conflict rolls no dice and compares no numbers. Outcomes are exclusively determined by what would be most entertaining for the participants.

Flaw: A selected characteristic that is specifically *not* also a gauge. A character either has a flaw or he does not. Flaws are structurally very similar to *gifts*. But, flaws are generally considered detrimental to a character rather than beneficial.

Fortune: An outcome that is at least partly based on random factors. This may include rolling dice, drawing cards, or some other random value generator.

Game Master (GM): A player assigned responsibilities different from other players. These responsibilities commonly include acting as the final authority in disputes, playing NPC's, describing scenes, etc. Some games have no Game Master. (See the Game Master design pattern.)

Gauge: A graduated value generally associated with a name. Commonly the graduated values are numbers, but this is not always the case. (See the Gauge design pattern.)

Gift: A selected characteristic that is specifically *not* also a gauge. A character either has a gift or he does not. In general, gifts are considered beneficial to a character's well-being. (See the Gift design pattern.)

Handicap: A selected characteristic that is also a gauge and is generally considered detrimental to a character's well-being. Handicaps are structurally similar to skills, in that the only major difference is that handicaps are detrimental to characters whereas skills are beneficial. From a design pattern perspective, virtually anything that can be said concerning a skill also applies to handicaps. So, the Skill design pattern applies to handicaps as well.

Karma: An outcome based on non-random value comparisons. A karma-based contest directly compares two values to determine an outcome.

Non-Player Character (NPC): A character portrayed by the Game Master as part of that role.

Optional Characteristic: A characteristic that is not common to all characters of a given type.

Player: Any person participating in a role-playing game.

Player Character (PC): A character portrayed by any player while not assuming the role of Game Master.

Primary Attribute: An attribute whose value is set directly by a player rather than being derived by a formula from other attributes. Commonly Primary Attributes are used in formulae to determine the values of Derived Attributes but their own values are not determined by formulas. Typically, primary attribute values are generated by die rolls or set by spending some resource.

Rank: The specific value of a gauged skill, handicap, or ranked trait. Also used as an adjective in place of *gauge* when describing such skills and traits (i.e., "Horsemanship is a ranked ability.") (See the Rank design pattern.)

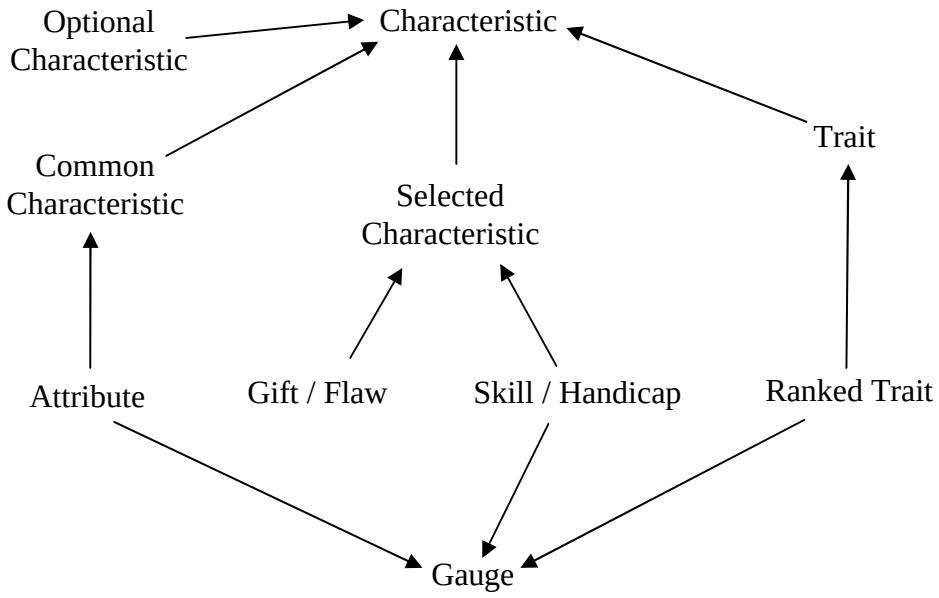
Ranked Trait: A *trait* that is also a *gauge*.

Selected Characteristic: A characteristic selected from a pre-defined list of choices.

Shared Gauge: A *gauge* that is shared by many characters.

Skill: A *selected characteristic* that is also a *gauge* and is generally considered beneficial to a character. (See the Skill design pattern.)

Trait: A *characteristic* made up by a player without drawing it from a pre-defined list of choices. (See the Trait design pattern.)

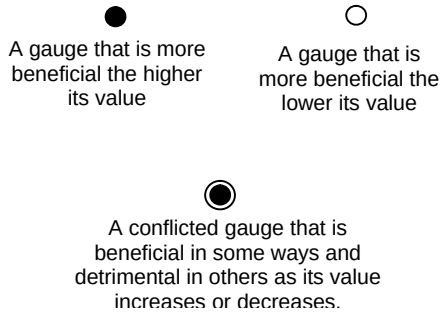


Arrows indicate "is a type of," so an Attribute is both a Common Characteristic and a Gauge

Gauge Diagrams

Gauge diagrams illustrate a game's core gauges and their relationships to one another. The purpose of creating a gauge diagram is to convey understanding of a game's currency flow from one game designer to another. So, a solo designer that neither needs nor wants outside assistance will find gauge diagrams to have little value. However, most of us crave feedback on our designs and gauge diagrams can assist us in obtaining it. The diagrams ease the learning curve for those from whom we seek advice by making a game's currency flow very clear.

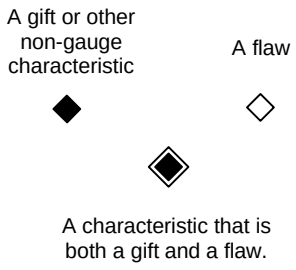
Most gauges consist of a name and an associated value. The value is mandatory, the name is not. That does not mean that the value of a gauge need be numerical, though. A game master's estimation of how well a player role-played in a session is a gauge, albeit a very subjective and fuzzy one. For example, a game master might evaluate a player's performance in a variety of ways, including any of the following: "Wow, you really rocked tonight!", "You seemed a little distracted, but things turned out all right", or "I don't think your heart was really into the game".



Representing Gauges, Gifts, and Flaws

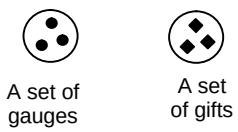
In gauge diagrams, nodes (circles) represent gauges. A filled circle indicates that big gauge values benefit the character more than small ones. An empty circle indicates that small gauge values benefit the character more than big ones. Some gauge values are conflicted. They neither benefit nor punish a character as their values change. Or, rather, they do both simultaneously (see the Conflicted Gauge design pattern). Such gauges should be diagramed with a filled dot within an empty circle.

Characteristics that are not gauges, such as classes, gifts, and flaws, can also be diagramed. These are represented as diamonds. Gifts are represented as filled diamonds while flaws are represented as empty diamonds. A characteristic that can be considered both a gift and a flaw is represented by a filled diamond surrounded by an empty diamond.



Representing Sets

At times, it is convenient to diagram a collection of gauges as a single node. This is done when the gauges of interest are treated uniformly in the game design. A set of gauges is diagramed as a circle containing dots or diamonds.

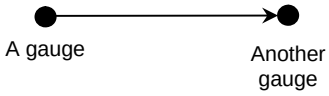


It is also sometimes useful to be able to diagram and reference an individual gauge that also happens to belong to a set of gauges. In such cases, both the gauge and the set are diagramed and the mathematical symbol ∋ (is an element of) is inserted between them.

A gauge that is an element of a set of gauges

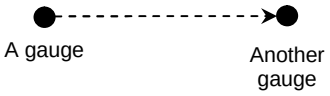
Representing Relationships

Arrows represent relationships between gauges.

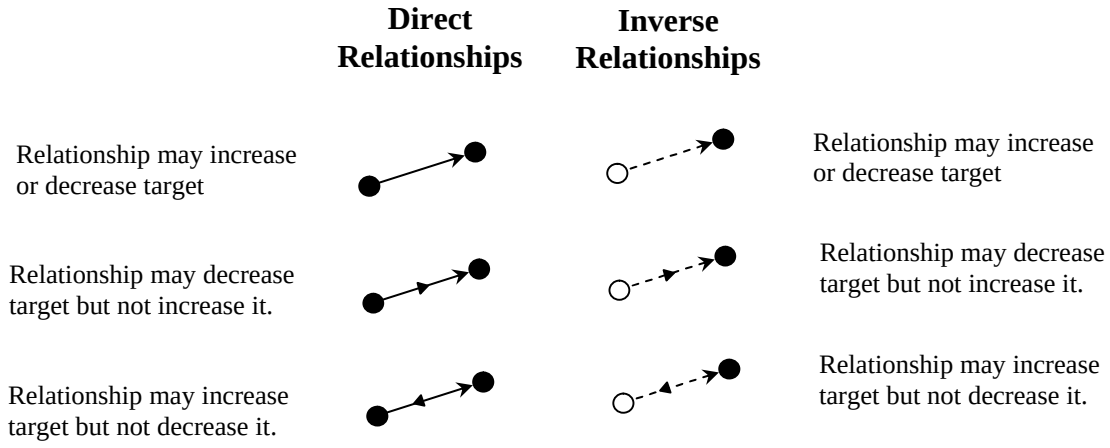


This diagram illustrates a relationship between two gauges. The solid line on the arrow indicates that the relationship is a direct relationship. That is, the targeted gauge value rises if the originating gauge value rises and/or the targeted gauge value decreases if the originating gauge value decreases.

A dashed arrow indicates an inverse relationship. That is, the targeted value decreases as the originating value increases and/or the targeted value increases as the originating value decreases.



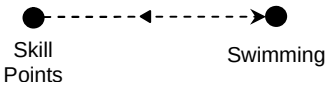
Some relationships both increase and decrease the targeted gauge. Other relationships can only increase the targeted gauge value or decrease it, but not both. The ambiguity of precisely how a relationship affects the target is resolved by placing adornments on the relationship. Small triangles are placed on the relationships to clarify the relationship's nature as shown in the following diagram:



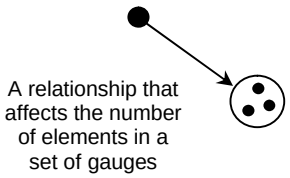
The triangles are placed on the lines so that you can read the diagrams easily. Just remember these simple rules:

1) If the base (big end) of a triangle points toward the target, the relationship may increase the target.
2) If the tip (little end) of a triangle points toward the target, the relationship may decrease the target.

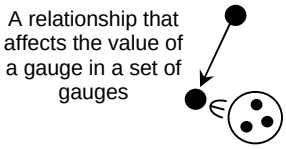
So, as you travel along the arrow from the origin to the target, if the triangle widens, it may increase the target. If it grows narrower, it may decrease the target. If no adornments appear on the relationship, then the relationship is assumed to do both.



A resource relationship is a good example of a relationship that requires an additional indicator. A resource is a gauge whose value can be "spent" in order to affect another gauge value. For example, a character may have a resource called "Skill Points" that his player may spend to raise the character's rank in the skill of "Swimming." This relationship might be diagramed as shown here. The relationship is an inverse relationship because Skill Points can be spent down to raise the character's Swimming rank. But, if the character later gains more Skill Points, his Swimming rank is not going to lower as a consequence. So, the triangle adornment on the arrow indicates that the relationship can increase the Swimming rank, but cannot decrease it.

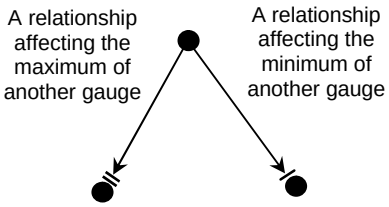


When a relationship points directly to a set, it does not affect the values of the elements in the set. Rather, the relationship illustrates that it affects the *number* of elements in the set. Such a relationship could be used to indicate that a resource could be spent to "buy" more skills or gifts.



When one gauge *does* affect the *values* of the gauges in a set of gauges rather than the *number* of elements in the set, the "element of" icon shows that a gauge is affected by a relationship and that it is also contained within a set. In the example diagram, if the upper gauge represented a character's "Level" and the set represented a character's Skills, the diagram would be saying that the Level gauge tends to generally increase the Skill ranks as its own value increases. It does not mean that the Level gauge necessarily increases *all* of the referenced Skill gauges or that it increases them equally, only that it tends to increase some of them.

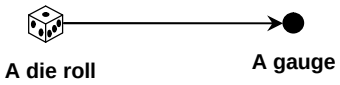
Illustrating Minimums and Maximums



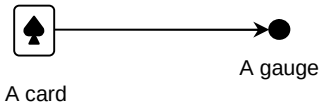
Occasionally, a gauge does not affect another gauge directly, but rather affects the minimum or maximum value that the other gauge may take. In such cases, the arrow representing the relationship does not point to the gauge itself. Instead, it points

to a single or double line placed next to the targeted gauge. A double line segment represents a gauge maximum. A single short line segment illustrates a gauge minimum.

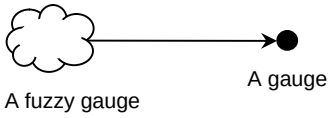
Special Icons



Die rolls are a very common gauge used in role-playing games. The diagrams do not use a simple dot or circle to represent a roll of dice (although we could, since a die roll is merely another kind of gauge). Note that the icon is that of a six-sided die, but it can represent the roll of any kind and number of dice. The icon represents a random number generation while abstracting away the details of exactly how that number is produced.



Sometimes, games use cards to generate random values. In such cases, a card representing the Ace of Spades is used instead of a die icon or simple dot. Again, this is purely for aesthetic reasons to increase the diagram's readability. It does not imply that a standard card deck is used, only that a card is drawn from some deck. In fact, role-playing games that use cards often have their own custom decks.



Subjective gauges, or gauges whose values are particularly fuzzy, are represented using a cloud.

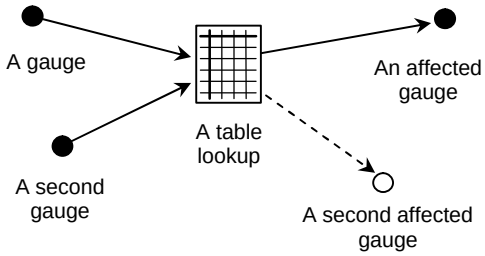
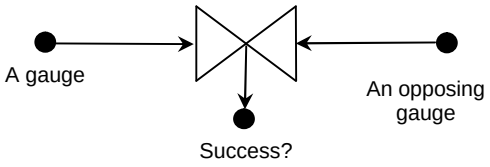


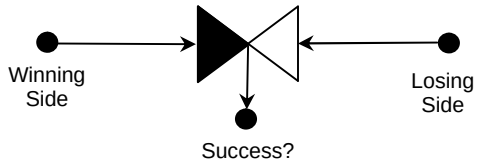
Table lookups are also represented using a special icon. Table lookups always have one or more gauges providing input and they have an effect on one or more gauges.

Contests



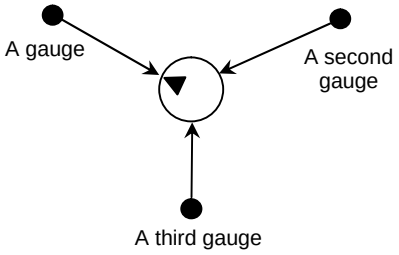
Contests pitting two forces against one another are represented as a pair of triangles joined at one tip. Contests always have two input nodes and affect one or more gauges as output. Often times, the output is a gauge that answers a question, such as whether a character succeeds in some action. At other times, a contest generates a *degree* of success.

If you want to distinguish the winning side from the losing side of a contest for some reason, fill in the winning side of the contest icon.

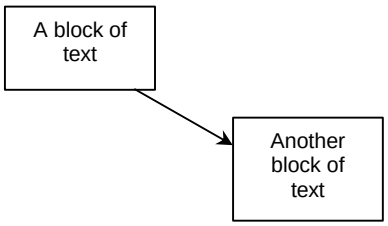


Sometimes, a game system chooses one option from a list, such as when determining which player has the right (or responsibility) to take his turn. Somehow, the system selects one player as the next in line. The icon representing this kind of contest takes the form of a circle containing an offset triangle. This represents a kind of "dial" that turns from player to player as their turns come up.

Text Blocks



Finally, sometimes we need to diagram actual blocks of text within the rules. These are represented as simple boxes enclosing the text. Quite often, one text block refers to another text block. When such references need emphasis, an arrow is drawn between the boxes to show the reference. In such cases, the diagram customarily contains a description of the relationship.



Design Patterns

To raise a “rule of thumb” to the status of design pattern, it must be formalized. Christopher Alexander stated that a design pattern must contain at least the following four aspects: a name, a problem statement, a proposed solution, and the consequences of using the pattern. The “Gang of Four” patterns have even more aspects, which are meant to further clarify the problem domain and proposed solution. The design pattern template used in this text mimics the Gang of Four format, with slight modifications appropriate for role-playing game design.

Pattern Name

If the pattern has a succinct and commonly known name, use it. Otherwise, make the name short and descriptive. A good name is crucial, because it becomes the definitive term used to reference the pattern in future discussions. Because design patterns are neither “appropriate” or “inappropriate” for a game without first knowing the designer’s goals, their names should be neutral rather than connote a value judgment.

Intent

Provide a short statement (one or two sentences at most) describing the rationale for the pattern’s existence. What does it accomplish? What problem does it solve?

Also Known As

Give any other common names used for referencing the pattern.

Related Patterns

Provide references to any similar patterns that should be considered as alternatives when this pattern is considered.

Motivation

Provide a detailed description of the problem being addressed by the pattern and its solution. This description may be as lengthy as necessary to describe the problem being addressed.

Example Structure (Optional)

Provide a diagram of the participants in the pattern. Most of the diagrams in this book are gauge diagrams. The diagrams abstract away details to expose the underlying structure.

Applicability

Provide a list of criteria of when to use the pattern. When can it be applied? What kinds of poor designs does the pattern address? What are the alternatives?

Consequences

Provide descriptions of the consequences of using the pattern, both good and bad.

Implementation Concerns

Provide descriptions of what practical design issues should be considered when applying the pattern.

Samples

Provide examples of using the pattern. Keep the examples as simple as possible to illustrate only the key points.

Known Uses

Reference games using the pattern and explain how they use it. This need not be an exhaustive list of *all* known instances of use, but rather a sampling of uses that covers the broadest possible spectrum of current application. There should be at least 2 examples in this section to ensure that the pattern is, in fact, a pattern and not just a single-use idea (however brilliant).

RPG Design Pattern Catalog

The following sections present a list of design patterns gleaned from the study. The design patterns are partitioned into several categories, including: Conflict Resolution Patterns, Character Makeup Patterns, Fundamental Gauge Patterns, Miscellaneous Patterns, Reward Patterns, Role-Playing Patterns, Story Patterns, and Structural Patterns.

Conflict System Patterns

Contest Tree

Intent

Provide a mechanical means to create rising tension within a game.

Also Known As

Escalating Conflict

Related Patterns

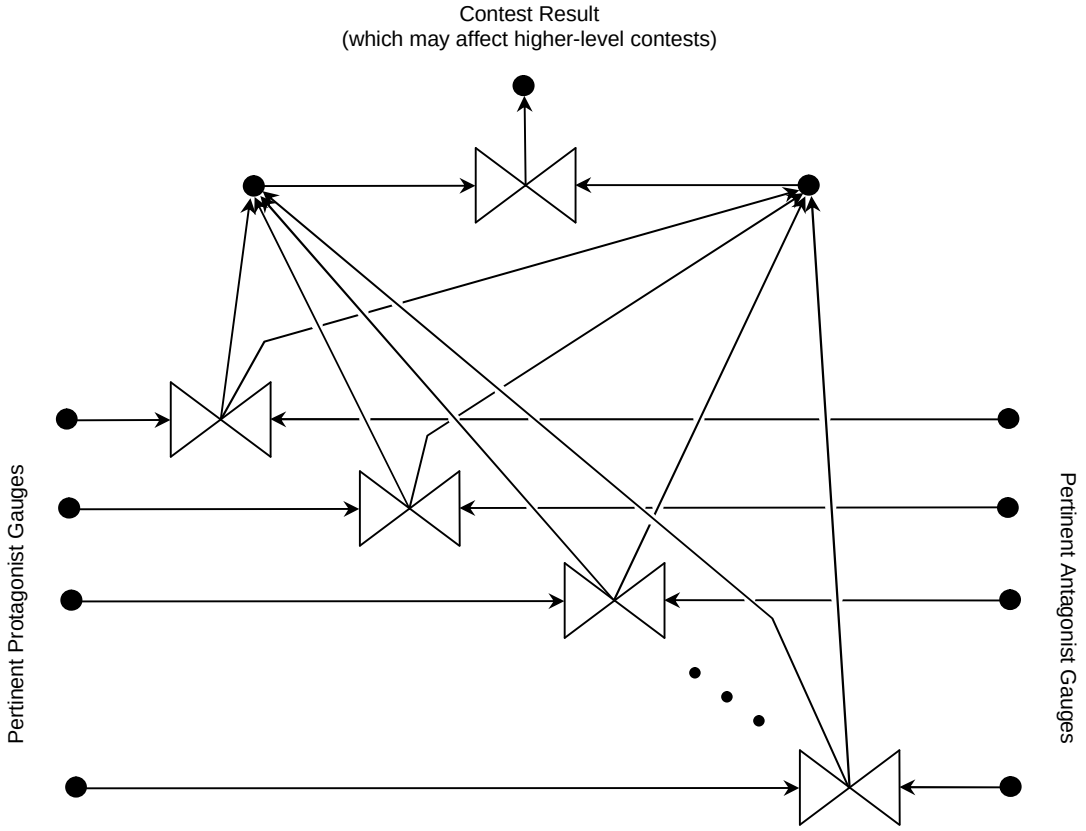
End Game, Generalized Contest, Last Man Standing, Negotiated Contest, Structured Story

Motivation

A Contest Tree is a high-level conflict resolution system made up of many contests arranged in a hierarchical fashion. The results of lower level contests feed into and affect the outcomes of higher level contests. Supposedly, players have an interest in the outcome of the higher level contests. Therefore tension arises concerning the eventual success or failure of the high level contests as lower level contests succeed and fail. If a game has a mechanical means for the results of lower level contests to feed into the outcomes of higher level contests, it follows the Contest Tree pattern.

Contest Trees exist in many games, the most traditional form of which is detailed in the Last Man Standing design pattern. However, it is interesting to note that none of the games studied as potential sources of design patterns used mechanical contests to resolve their highest-level conflicts (which would be illustrated by whether Luke Skywalker is killed or convinces Darth Vader to turn back to the Light Side and overthrow the Emperor; or whether Indiana Jones finally obtains the Lost Ark or loses it to the Nazis). All studied games depend on GM or player Fiat to resolve these issues. Therefore, this is fresh territory that new game designs could explore.

Example Structure



Applicability

Use the Contest Tree pattern when you want to create a sense of rising tension in your game as play progresses and don't mind using a mechanical means of accomplishing this goal. If you want to keep your game as free as possible from gauge mechanics but want to create a sense of dramatic tension, then you should put sufficient effort into giving players instructions on how to do this. Generating suspense does not come naturally to everyone and is an important part of game play.

It is certainly possible to generate suspense in a game by other ways. For example, a skilled Game Master can create tension by the way he narrates scenes and manipulates circumstances. However, tension simply *cannot* be created without conflict of some sort and some games can certainly be improved by introducing mechanics that support its generation. A Contest Tree is one such means.

Consequences

The Contest Tree pattern does inject a sense of anticipation into a game, provided the players are invested in the high-level goals of the overall conflict. The drawback is that it also introduces artificial mechanisms to make this happen. Some gamers prefer little or no artifices in their games to jar them from their immersion in the game world.

Implementation Concerns

A few key points to creating tension (whether you are using a mechanical means of doing so or not) are:

1) The protagonists and antagonists should be evenly matched.
2) The protagonists and antagonists should both periodically fail in their attempts, proving they are worthy adversaries.
3) The protagonist and antagonist successes and failures are never so great that all hope of success of attaining the high level goals is eliminated from either side too quickly. (A quick and total victory for either side demonstrates the protagonists and antagonists were mismatched somehow. This may occur because the characters' abilities are grossly unequal, or because the game mechanics improperly implement the Contest Tree pattern.)

If either side of a story's over-arching conflict is obviously going to win with no hope for the opposing side, tension cannot arise. Sports events are prime examples of this. If one sports team is obviously outmatched by another, the game is predictable and boring. This does not mean that both sides must be evenly matched in similar respects, such as personal power.

A good literary example of evenly matched protagonists and antagonists that *appear* mismatched is found in The Lord of the Rings trilogy. Despite appearances to the contrary, Frodo Baggins and Sauron are evenly matched adversaries. If you are familiar with the series (and what RPG'er isn't), Frodo is obviously outclassed by Sauron in terms of raw power. If Sauron knew where to find Frodo and the One Ring, Frodo would have no hope of success in his quest to destroy the ring. But, Frodo's selfless tenacity, the secrecy of his location, and Sauron's inability to find him make the two characters evenly matched. Sauron's continual and credible efforts to find Frodo and his periodic successes in *almost* capturing the ring create tension throughout the story. Frodo's eventual success in throwing the One Ring into the volcanic fires of Mount Doom proves that both are worthy adversaries.

The same kind of meek hero vs. overpowering villain conflict can work in a game, provided you give each opposing side equal control over the developing storyline. Note that this would be difficult if the primary means of influencing the plot in your game centers around the personal power of the various characters. If you want to create a game that generates these kinds of stories, you would be best advised to find some other means to resolve conflicts. One possibility is to introduce a renewable resource that can be spent to directly affect conflicts. (Universalis takes this approach to resolving conflicts, although it doesn't implement the Contest Tree pattern.)

You don't necessarily have to use the same Negotiated or Generalized Contest design pattern to resolve all contests at all levels. It is quite possible to mix and match different resolution methods at different levels. For example, you could use Generalized Contests to decide the lowest level conflicts but use Negotiated Contests on higher level conflicts. Of course, it is also possible to use the same resolution method at

all levels, provided you design it that way. (Using the same resolution technique at all levels would mimic recursion in software design, if that has any meaning to you.)

Most sports events use Contest Trees to determine who wins and can provide inspiration for ways to do this in role-playing games. For example, you might decide to use a "Beat the Clock" type of system as is used in American Football, where opponents try to accumulate the greatest number of points within a set time period to determine a winner. Or, you might use an "Inning" technique as is used in Baseball where the conflict is split up into a set number of innings in which the antagonists and protagonists take turns trying to earn points toward eventual victory. Tennis uses a hierarchical scoring system where the players accumulate Points to win Sets and Sets to win Matches. The professional versions of all of these sports have even higher level contests. Every season builds up to who wins the Superbowl, the World Series, or Wimbledon.

One other issue should be highlighted concerning Contest Trees. It may seem obvious once stated, but a Contest Tree can only resolve the kinds of high-level conflicts dealing with the kinds of mechanical inputs provided by the system. In other words, if "Damage" and "Remaining Hit Points" are the only gauges used as input into a conflict resolution mechanic, then that mechanic can only resolve issues dealing with Damage and Hit Points. Creating a game that is flexible in the kinds of conflicts it can resolve equates to designing flexible conflict inputs as well as outputs.

Samples

Let's suppose that we want to design a narrative game that supports the creation of dramatic tales. We want the stories to have a sense of rising tension and we want to have our system support this. We decide to break the stories down into an indeterminate number of Acts; break Acts into an indeterminate number of Scenes; and break Scenes down into a number of Actions.

Before play beings, we'll have the players design both the protagonist and antagonist characters that will appear in the story. They must also decide upon the overarching conflict that exists between them, including negotiating what happens if the antagonists win and what happens if the protagonists win. Similarly, at the beginning of every Act, we have the players negotiate what sub-conflict is being resolved within the Act to bring the overarching conflict closer to resolution. Then, at the beginning of every Scene within that Act, we have the players negotiate what conflict the scene resolves that brings the Act's conflict closer to resolution. Finally, we have the characters perform individual actions. Just for grins, we'll use Generalized Contests to determine the results of Actions. So, for every contested action we'll have the player roll a d6, add pertinent gauge values for the acting character, and compare the result to a threshold based on his opponent's appropriate gauges. The first side (protagonist or antagonist) to attain 3 Action successes within a Scene wins the right to narrate the outcome of the Scene's conflict. The first side to attain 3 Scene successes wins the right to narrate the outcome of the Act's conflict. Finally, the first side to attain 3 Act successes wins the

right to narrate the outcome of the overarching conflict. All such narrations must conform to the terms agreed upon during the conflicts negotiations.

Known Uses

Dogs in the Vineyard has an escalating conflict resolution system that follows the Contest Tree design pattern. Characters on opposing sides of a conflict each have dice pools whose number and type are set by the character's attributes and traits. Different attributes are brought into play depending on actions taken. The dice are rolled and the aggressor selects two of his dice and puts them forth as a "Raise" (whose value equals the sum of the two dice). The defender must then meet or exceed this value by putting forth dice of his own to "See" the Raise. If he can See with a single die, he has attained a "Reversal" allowing him to re-use that die in his next Raise. If he can See with two dice, neither side suffers any effects. If he requires three or more dice to See, he suffers "Fallout" (wounds and other effects). If a player feels he is losing the conflict, he can decide to "Escalate" it by having his character pull a knife or gun, which allows him to add the dice associated with the action to his dice pool. Of course, his opponent can do likewise. The person winning the last exchange in the conflict wins the overall conflict. The system contains low-level conflicts consisting of Raises and Sees that decide a higher level result. The low-level conflicts' Fallout also has an effect on subsequent conflicts. Optionally, the Game Master may ignore the Fallout inflicted on an NPC and, instead, hand the Fallout dice over to the protagonists to use in upcoming conflicts. How all of this affects an adventure's overarching goals, though, is not obvious from the text. Supposedly, if the protagonists accumulate enough success, the GM will eventually decide the players attain what they are striving for.

Dungeons and Dragons v.3.5 uses Generalized Contest for its lowest-level contest system. It is highly combat oriented and the main goal of the system is to deplete the opposing force's available resources before your own resources are completely drained. Individual character actions are almost always geared toward this goal. For example, a sword swing delivers damage, which detracts from Hit Points (see the Hit Points pattern), one of the primary resources in the game. When the protagonists deplete an individual antagonist's resources, they gain victory over that character. (There are other ways to gain victory over a character, but this is the primary means.) When the protagonists gain victory over all antagonists in a scene, they gain victory in that battle. When a sufficient number of battles have been won, the protagonists fight their way to the final battle of the adventure. If they defeat the adventure's primary villain(s) in this climactic battle, they win whatever goal it was they sought by setting out on the adventure (this goal may or may not have been negotiated with the Game Master prior to starting the adventure). So, D&D follows the Contest Tree design pattern with three layers:

1) Individual character actions (Generalized Contests) feed into
2) whether an individual character is defeated (also Generalized). Victories over characters feed into
3) whether a battle is won or lost (also Generalized).

This particular form of Contest Tree is quite common and is described in depth in the Last Man Standing pattern. The highest layer conflict, if it exists, does not fall into the Contest Tree pattern, because it is not an actual mechanical contest. It is a conflict determined by GM fiat that the characters have won "enough" battles to win overall victory. Hopefully, all this feeds the players' thirst for more of the same.

Generalized Contest

Intent

Provide a mechanical means to resolve disputes where the possible set of outcomes is negotiated in advance of a conflict's introduction.

Also Known As

Task Resolution

Related Patterns

Contest Tree, Negotiated Contest

Motivation

The Generalized Contest design pattern structures contests is a way where the possible set of outcomes is negotiated among the players before a conflict is even introduced. Generalized Contests accomplish this goal by making available a set of allowable actions that a character may undertake along with the possible results of those actions. That is not to say that a game must specify all actions that a character may undertake, only that those actions that are resolved through Generalized Contests must specify their set of possible outcomes in advance of a conflict's introduction. These "pre-negotiations" essentially become part of the group's "Social Contract", the unspoken but mutually understood code of acceptable behavior and other ground rules that evolves within any cohesive group. Their adoption may be as simple as all players agreeing to follow the rules as specified in a game's text where the game provides a list of Skills along with their effects. Any alterations or re-interpretations of those rules demand a re-negotiation of the Social Contract. In this way, house rules arise that adjust the rules as stated.

The specification of possible outcomes may thus come in any of the following forms:

1) Game text describing the possible set of outcomes of specific actions, such as individual Skill descriptions. (i.e., "The skill of opening locks can be used to unlock a single lock on a successful roll.")
2) Game text describing actions in a general way, but specifying in detail the set of possible results of actions. (i.e., "An individual action cannot resolve a higher-level conflict, but it can result in any of the standard side-effects (player choice).")
3) Prior player agreement that adopts a house rule into the Social Contract.

The important difference between this pattern and the Negotiated Contest design pattern is that the outcomes of Negotiated Contests are negotiated by the players after a conflict is introduced while those of Generalized Contests are negotiated before a conflict is introduced. Because Negotiated Contests require negotiation, they are highly flexible and can adapt to virtually any kind of conflict. Generalized Contests lack this flexibility

because they require generalizations that do not always fit the situation at hand. On the other hand, Generalized Contests have the advantage of speed (on a contest-by-contest basis) in that all negotiations have taken place prior to a conflict (usually far in advance of the conflict).

A game using Negotiated Contest mechanics to decide contests can actually evolve into one commonly using Generalized Contests through play. This happens when certain character actions become so common that the Social Contract of the group accepts that certain character actions always result in one of a small set of outcomes. Such cases occur when players describe actions that others accept without question because the validity of similar actions has been previously established through negotiation. Since the negotiation pertaining to the action has already transpired, arguing against its validity might violate the Social Contract without a re-negotiation. A debate opened up on a character action might result in an alteration to the Social Contract so that a new set of pre-defined outcomes is established for a class of action. The reverse is also true. An action that would normally be resolved through a Generalized Contest might occasionally be negotiated to take into account special circumstances. Because of the malleability of Negotiated and Generalized Contests, the patterns are mirror images of one another.

Applicability

Use the Generalized Contest pattern when you:

1) Want to resolve some (possibly most) in-game conflicts without a negotiation phase.
2) Are willing to resolve conflicts that cannot be handled in a general way through Negotiated Contests.

The pattern is especially applicable to tactical games where players pit their gaming skills against one another. The pre-defined nature of Generalized Contests allows applicable conflicts to be resolved in an unbiased fashion. A big part of any such game revolves around player knowledge of the available tactical options and their skill in taking advantage of them.

Generalized Contests are also appropriate for games where the design goals include creating lists of pre-defined Skills, Gifts, Flaws, and/or Handicaps. What this essentially provides players is a collection of (hopefully) well thought-out generalized win/lose conditions for a wide variety of character actions. Thus, a large body of quality conflict material can be easily adopted by the simple act of consenting to view the game text as authoritative.

Note that this consent is itself a negotiation among the players that adopts the game text into the group's Social Contract when playing the game. If some players later decide that they don't like some of the game rules as stated, they can either

1) Re-negotiate the generalized set of possible outcomes for specific actions and adopt these as house rules, or
2) Negotiate the set of possible outcomes for a single conflict after it has been introduced into play.

In the first case, the negotiation merely replaces an old generalized rule with a new one. So, it does not transform a Generalized Contest into a Negotiated Contest. However, the second case does transform one into the other. So, you would be wise to explicitly include some rules in your game to cover the possibility.

Consequences

When negotiation enters the picture, the inherent subjectivity of the process can leave some players feeling cheated in tactical games when they lose. Hard feelings between players can result.

Generalized Contests quickly resolve common, often repeated conflicts. They also allow a game author great influence over his setting and the kinds of conflicts that will arise in his game. He is afforded this opportunity by virtue of the fact that he can include as part of his game text descriptions of pre-defined contests in the form of Skills, Gifts, and the like. These kinds of descriptions also provide players with myriad examples of how the author envisions the game to be played.

If overused, the pattern can result in excessive amounts of text to cover all eventualities (aka "rules-bloat"). It is probably a mistake to rely exclusively on Generalized Contests to resolve all conflicts. This pattern is incapable of handling all possible special cases that can arise in tabletop role-playing games. So, trying to force-fit the pattern to handle more and more circumstances will only result in an ever-expanding set of rules covering a never-ending stream of special cases.

Implementation Concerns

In games where Negotiated Contests are not available, characters can attain goals only through pre-negotiated rules. If a specific desired outcome is not adequately covered by a rule, a problem arises. For example, the following exchange could occur in an anthropomorphic game:

> *Game Master:* "A fierce thorn-wielding chipmunk with engorged cheeks blocks your frog's path."
> *Player:* "Grenouilles engages the rascal with a haughty croak."

Depending on what the player wants to accomplish, a Generalized Contest may or may not be appropriate. Suppose the player states his goal as follows:

> *Player:* "I punch the chipmunk in the stomach to make him spit out the purse he stole."

It is unlikely that any game's Generalized Contests include the possibility of forcing chipmunks to spit out purses. The questions that must be answered by the system may be something along these lines:

1) Does Grenouilles punch the chipmunk in the stomach?
2) Is the punch sufficiently forceful to cause the chipmunk to spit out what he has in his mouth?
3) Does the chipmunk actually have the purse in his mouth, or is it something else?

The first two contests may reasonably be covered by a game's collection of Generalized Contests. But the third question is unlikely to answerable in such a fashion. Many games give the Game Master the responsibility of deciding these kinds of questions. But, therein lay a problem. To put it simply, such a system gives players no real way of accomplishing goals that fall outside the game's set of Generalized Contests. Instead, they must rely purely on the Game Master to give them what they want. In other words, there is no negotiation phase taking place before a contest, so a successful outcome might or might not result in what a player desires. So, our previous example might end as follows:

> *Game Master:* "With a loud 'Ooof,' a shower of *acorns* spews out of the chipmunk's cheek pouches."

In a game without negotiated contests, a successful outcome to a contest does not equate to the player gaining what he wants. A Game Master might unilaterally decide to alter the actual effect. Similarly, a failed outcome does not equate to the player losing, either:

> *Game Master:* "The chipmunk deftly avoids your attack and retains the contents of his mouth. But, you notice the glint of gold under a nearby bush."

So, in a game lacking negotiated contests, the determination of the success or failure of a character action has no absolute bearing on whether a player actually succeeds in his goals. To avoid this problem, you can either

1) Implement rules that allow a Negotiated Contest to replace a Generalized Contest when appropriate, or
2) Implement rules that allow the results of your game's Generalized Contests to feed into and influence higher-level Negotiated Contests. (See the Contest Tree design pattern for advice on how to do this.)

If the game incorporates Negotiated Contests, the Game Master and the player would negotiate the actual effects of a successful punch prior to rolling any dice (or drawing cards, comparing numbers, or whatever). Such a negotiation might be as simple as "Okay, if you win, then he'll spit out the purse. If you lose, then Grenouilles trips, allowing the chipmunk to scamper off into the underbrush. Is that acceptable?"

Assuming the agreement is acceptable to both players, a successful outcome would result in something like the following:

> *Game Master:* "With a loud 'Ooof,' a shower of coins spews out of the chipmunk's cheek pouches. With a gasp and a wheeze, the purse follows shortly thereafter."

Samples

Let's create a simple Generalized Contest system for a game using skills.

The Skill Roll

A Skill Roll tells you whether your character can successfully use one of his skills. For example, you will make a Skill Roll when your character tries to pick a lock, sneak unheard, or train animals. To make a Skill Roll

1) Roll a d30.
2) Add your character's pertinent skill rank and appropriate attributes to your d30 score.
3) If the total equals or exceeds a set threshold, your Skill Roll succeeds. Unless otherwise stated, the threshold that any Skill Roll must overcome equals 15 plus the skill rank of any opposing agent.

The effects of success and failure for the most common uses of a skill are given in the skill's description. However, at times you will come across situations in which using a skill seems appropriate to solve a problem, but the game text does not take into account the specific circumstances facing your character. In such cases, it is perfectly reasonable for you and your Game Master to discuss the situation and determine any adjustments that need to be made to the roll. Perhaps more important, though, are that you and your Game Master feel comfortable in negotiating the possible effects of success and failure for your current situation. The effects of success and failure can be wide ranging from those stated in the text.

Known Uses

The Game Summaries section has extensive coverage of various Generalized Contest systems, and it would be pointless to repeat them here. You should pay particular attention to *Dogs in the Vineyard* (whose generalized "Raise" and "See" mechanics feed into higher-level negotiated contests), and *Rifts* (to see how overusing Generalized Contests can result in rules-bloat).

Last Man Standing

Intent

Provide a Generalized Contest Tree to resolve which side attains victory in battle.

Also Known As

Not Applicable

Related Patterns

Contest Tree, Generalized Contest, Hit Points

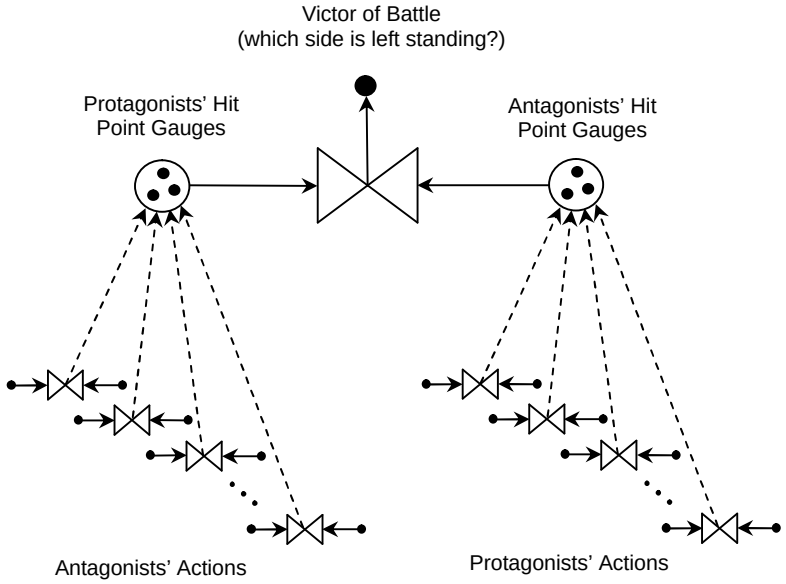
Motivation

The Last Man Standing design pattern is one of the oldest, most traditional forms of Generalized Contest Tree in existence. Its roots go all the way back to the pre-role-playing era of war-gaming. The pattern's intent is to provide a means to judge which side of a battle is the victor. Even though many combat-oriented games go into exacting detail concerning how individual characters can be defeated, they often leave who attains overall victory of melee implied. Obviously, the side that manages to avoid being entirely beaten to a bloody pulp wins.

The Last Man Standing design pattern does not demand that individual characters defeat one another using the Hit Points pattern. For example, a player could have his character drug all of his opponents to render them helpless. However, the combination of the Hit Points and Last Man Standing patterns is quite common.

Example Structure

The following diagram illustrates how individual character actions affect the Hit Points gauges of the opposing side. The first side having all their combatants fall due to damage loses.



Applicability

Use the Last Man Standing pattern when your game has a strong emphasis on tactical combat. If you want to resolve high level conflicts with means other than warfare, you probably want to avoid this pattern. You might want to consider adopting a more flexible Negotiated Contest Tree instead. If combat is a central focus of your game, but you also want to allow players to occasionally resolve conflicts by means other than bloodletting when they choose to do so, you may still decide to use this pattern. If so, you'll need to provide adequate text in your game explaining how the win/lose stakes of high level contests can be negotiated when needed. You'll also have to explain how the contest can be won independent of battle.

Consequences

If the only way to resolve high-level contests is through battle, then players will focus their efforts on being the best they possibly can at winning battles. In other words, if a game only provides a single tool to resolve disputes, then you can be assured that players will become very adept and focused on using that tool.

In addition, this pattern has been done-to-death in many games. While there is nothing inherently wrong with using this pattern when appropriate, you might want to avoid its use simply to distinguish your game from others.

Implementation Concerns

Assuming your game has adequate rules covering how a "man" can be made to be "not standing," the Last Man Standing pattern is trivial to implement. It is so trivial, in fact, that most games following the pattern do not even give its readers the benefit of actually stating its existence despite its importance. It is best to be explicit about such things,

though. If you are going to use this pattern, it behooves you to tell your readers that the side whose characters all fall first is generally considered to be the losing side.

Samples

Suppose we want to want to create *Blood and Honor*, a game set in the Vietnam War where battles are lethal and where victory entails killing as many enemies as quickly as possible. A battle's victor is based purely on who remains standing at the end of the fight. To make it more interesting, though, we also want to explore the toll killing takes on the human psyche (and to explore how Last Man Standing can be spiced up a bit). Sometimes, we want the player decision to *not* kill to be a rational choice even though that conflicts with the goal of victory.

We'll give each character a "Wounds" attribute that follows both the Trauma Gauge and Hit Points patterns. If a character's Wounds value exceeds 6, he dies. Characters will also be given two more attributes: "Vengeance" and "Compassion," whose values range from 0 to 6 and which begin set at 2.

Contests are performed by rolling a number of d6. Each d6 rolling 4 or higher is counted as a success. The number of d6 rolled in a contest depends on the nature of the contest. When a character defends his own life or the life of another platoon member, he gets to add both Vengeance and Compassion to his dice pool. When a character attempts to kill an enemy when his life or the life of a platoon member is not directly threatened, he adds only Vengeance. If the character attempts to perform some non-combat action to save his own life or the life of a platoon member, he adds only Compassion. The number of successes in any combat action indicates the number that is added to the enemy's Wounds attribute. The Wounds value is always subtracted from that character's dice pool in all actions. Vengeance and Compassion are affected according to the following table:

Situation	**Affects (all options are player choice)**
Killing while Defending Comrade or in Self-Defense	+1 to Vengeance or Compassion
Killing in Cold Blood	+1 to Vengeance or -1 to Compassion
Resisting Killing when Killing is an option	-1 to Vengeance or +1 to Compassion
Saving a Comrade's Life without Killing	-1 to Vengeance or +1 to Compassion

No choice can raise an attribute above 6 or below zero. If this would happen, the other choice must be taken. A character's Vengeance and Compassion values may trigger dramatic events as follows:

Values	Effect
6 Vengeance, 0 Compassion	**Enraged:** Rage fills the character, who finishes the current battle without heed to safety. All actions strive to kill enemies.
0 Vengeance, 6 Compassion	**Pacifist:** Character will not attack except in self-defense.
0 Vengeance, 0 Compassion	**Catatonic:** Character blocks out reality and permanently refuses to perform any further actions.
6 Vengeance, 6 Compassion	**Bi-Modal:** Character permanently refuses to attack except in self-defense. But, when attacked, becomes *Enraged* as above.

While the game follows the Last Man Standing pattern and victory in battle always demands the killing of enemies, killing is not always a good idea on a personal level.

Known Uses

Dungeons & Dragons v.3.5, HARP, and **Rifts** all use Hit Points to gauge whether an individual character is mobile or incapacitated. They all implicitly incorporate the Last Man Standing pattern.

Negotiated Contest

Intent

Provide a mechanical means to resolve disputes where the set of inputs and possible outcomes is negotiated by the players specifically for the conflict.

Also Known As

Conflict Resolution

Related Patterns

Contest Tree, Generalized Contest

Motivation

The Negotiated Contest Design Pattern strives to provide a mechanical means of resolving conflicts while retaining the great flexibility promised by role-playing games. It does this by first recognizing that role-playing is first and foremost a conversation between individuals who are constructing an imaginary world in which mutually agreed-upon events transpire. One person introduces an idea that he thinks would be exciting or interesting, and if the others in the group agree to that idea, then that idea is accepted as having taken place. When one person wants to introduce something into the game world but another player wants something else, a conflict arises. The Negotiated Contest design pattern allows players to negotiate a collection of possibilities that are acceptable to all involved and then determine through some mechanical means which possibility is actually incorporated into the game world.

Depending on the game, the Negotiated Contest design pattern can handle challenges with broad strokes or fine detail. What is vital to Negotiated Contests, and what distinguishes them from Generalized Contests, is that Negotiated Contests always demand that the win/lose stakes be negotiated after a conflict is introduced but before any dice are rolled (or cards drawn, numbers compared, etc.) Often, modifiers affecting the odds of winning or losing are also negotiated. In Generalized Contests, win/lose effects are negotiated in a general way before a conflict is introduced.

Applicability

The Negotiated Contest design pattern requires that players be allowed to negotiate the effects of success and failure before the conflict is mechanically resolved. Now, a game can be written where the mechanical effects in some (possibly most) cases are clearly spelled out in the rules and so negotiation does not need to take place in these circumstances (i.e. a game may primarily use Generalized Contests). However, situations will inevitably arise where a player has goals that are not covered explicitly in the rules. At these points, the Negotiated Contest design pattern essentially demands that players negotiate the win/lose effects.

Use the Negotiated Contest pattern when your design goals include one or more of the following:

1) A desire to unambiguously decide whether the outcome of a contest means a player wins or loses his stated goals rather than whether or not his character succeeds in performing discrete actions.
2) A need to scale the resolution of contests to levels of granularity different than that of individual actions.
3) A willingness to allow players a great deal of narrative freedom in describing the results of contests, both good and bad.
4) Implementing your RPG as a video game on a computer is a low priority.

If your game is more about winning contests through skillful tactics and clever use of resources than about generating good narration with a focus on story, you might want to consider using Generalized Contest as your primary means of conflict resolution. Even so, you ought to consider allowing Negotiated Contests for those special cases that just are not covered by explicit game rules.

The negotiation phase of the Negotiated Contest pattern makes it potentially scalable to virtually any conflict. For example, the players may negotiate the outcome of an entire war based on a single die-roll. Thus, from the perspective of "getting it over with," Negotiated Contests have the ability to completely abstract away fine details in which players are uninterested. In this way, Negotiated Contests soundly defeat Generalized Contests in terms of *overall* speed, even though *individual* contests are faster without negotiation. However, as the Contest Tree design pattern explains, abstracting away low-level contests eliminates any mechanical means of creating rising tension in higher-level conflicts. If you want to engender a feeling of increasing anticipation as characters work toward a goal, resolution must be spread out over several contests whose outcomes somehow feed into the outcome of higher-level contests. One way to do this without demanding a negotiation phase in every low-level contest is to use Generalized Contests on the lowest-level contests and Negotiated Contests on higher-level contests. This gets you the best of both worlds.

Consequences

Negotiated Contests afford players great flexibility in the kinds of effects their characters can produce. Players also gain more control over the flow of a story, because a Negotiated Contest allows a player to specifically state his goals and make a roll to attain them rather than manipulate his character through numerous actions that may or may not result in achieving the goal. Therefore, a player does not have to defer so much to a Game Master to learn if a successful roll means his character actually accomplished what he set out to do.

On the other hand, Negotiated Contests are more time consuming on a per-contest basis because of the negotiation phase. A well-designed scalable Negotiated Contest system can overcome this problem by abbreviating a conflict into fewer rolls. As play

progresses, a game containing nothing but Negotiated Contests tends to evolve into one containing many Generalized Contests as players adopt the experience of previously negotiated contests as "house rules". So, players become more and more attuned to what kinds of outcomes the group as a whole will accept for specific actions. Negotiation, then, becomes unnecessary for a wide variety of situations.

Implementation Concerns

Although the negotiability of win/lose stakes for negotiated contests can be implied rather than explicitly laid out in the text, it is best to clearly spell out the fact. You might personally believe that the negotiability of stakes is obvious, but that perspective is far from universal. Many gamers are used to games that do not involve any form of negotiation prior to rolling dice. Since this single concept can have a big an impact on your game, it is better to make the process explicit rather than risk losing players because they cannot understand how to play effectively.

Negotiated Contests are a great deal more than just rolling some dice to see who wins. Game designers incorporating them into their games need to consider many issues to ensure they have covered all of the factors. Conflicts can be broken down into four phases. These phases have been given the names of *Intent*, *Initiation*, *Execution*, and *Effect* by Forge-ites.

Intent refers to the point at which a player states what his character is going to do, but where that action has not yet been incorporated into the game world. This phase is essentially a negotiation phase where the player's intentions can change as new facts are brought into the picture:

> *Player:* "I draw my rapier and kill the Merchant Prince!"
> *Game Master:* "Ummm. As you reach for your sword, you notice about a half dozen men standing in the shadows with crossbows pointing at you."
> *Player:* "Oh. Well, that changes things. I smile at the Merchant Prince instead and give him the letter from the King."

The player revised his initial statement because of the Game Master's intervention. Obviously, neither the player nor Game Master believed the stated action to have actually transpired yet.

Initiation is the phase at which a character action is introduced into the game world. Miscommunication of what is Intent and what is Initiation is a big potential source of arguments in games. After all, when a player says, "I draw my rapier and kill the Merchant Prince!" does he really mean that his character's attack has transpired regardless of circumstance? The ambiguity can result in disastrous consequences for a character and angry debate between players. In designing your game, you should seriously consider incorporating a formal negotiation phase to establish player intent before accepting anything as actually having been initiated by characters in the game world.

Execution is the phase in which the success or failure of a character action is determined. It is at this point that players roll dice, draw cards, or compare gauge values.

Effect is the phase in which the results of a character action are determined. When a player says, "I draw my rapier and kill the Merchant Prince!" does he really mean that success in his action means the Merchant Prince is lying at his feet dead? Or, is he merely conveying the fact that he is attempting to bring the Merchant Prince as close as possible to that condition? To avoid even more arguments, your rules must be clear on the extent to which character actions can affect the game world.

Not surprisingly, game designers have discussed the topic of Negotiated Contests from a wide variety of angles. In one discussion on The Forge website (www.indie-rpgs.com), Vincent Baker boiled a lot of that knowledge down to a checklist for game designers. (Vincent is the author of *Dogs in the Vineyard* and *Kill Puppies for Satan*, among others.) Vincent's checklist is paraphrased below, with slight alterations to terms and formatting to more closely align with those of this book. When designing your Negotiated Contest system, you should answer the following questions:

1. What does the winner get?
2. What does the loser get?
3. How do we know who's the winner and who's the loser?
4. What do we need to establish before resolution begins, and how?
5. What must we leave undetermined, for resolution to decide?
6. How do I know when it's my turn to say what happens?
7. When I say stuff, other people will probably have said stuff before me:
    a. What must I absolutely accept and take into account?
    b. What am I allowed to question, interpret, or change before I take it into account?
8. What should I establish for the next person?
9. What should I establish but leave open to the next person's interpretation?
10. What must I leave undecided?
11. When I say what happens, what subject(s) are available to me?
12. When have I said enough?
13. How much in-game action should I talk about?
14. How will I know not to say too little or too much?
15. During all this, when do I roll dice (or draw cards or whatever)?
16. After I've rolled, what decisions can I make about my dice?
17. If I don't like my roll, what are my options?

18. Some of the above decisions will depend on my roll. For each, does the result of my roll
    a. limit me to one option only?
    b. limit me to one of a few options?
    c. provide me with one or a few restrictions but otherwise leave me open?
    d. provide me with options otherwise unavailable?
19. Some of the above decisions will affect this roll or future rolls. For each, does what I say
    a. change an existing roll, mine or someone else's?
    b. change a future roll, mine or someone else's?
20. If such changes are open to interpretation or customizable,
    a. what are the options, and
    b. who gets to choose?

Samples

Let's create an ultra-simple system for Negotiated Contests. Whenever a conflict arises between two players, they perform the following actions:

1) Discuss the in-game situation and attempt to negotiate a mutually agreeable outcome.
2) If no mutually agreeable outcome can be found, the players should negotiate two possible outcomes, one that happens if one player wins the conflict and another if the other player wins. The negotiations should include any limits placed on the winner as to how he can describe the outcome.
3) One of the players tosses a coin into the air. The other player calls it as either "heads" or "tails."
4) The winner of the coin toss wins the conflict and can narrate the outcome within the agreed-upon limits.

Known Uses

The Game Summaries section has extensive coverage of various Negotiated Contest systems and there is no need to repeat them here. You should pay particular attention to *Code of Unaris* (for "hacking"), *Dogs in the Vineyard* (for escalation mechanics), *Donjon* (for player introduced facts), *My Life with Master* (for scene-level resolution), *The Pool* (for its gambling mechanics), *Sorcerer* (for how currency affects conflicts), and *Universalis* (for GM-less conflict resolution).

Safety Valve

Intent

Provide a scarce resource to players that they can spend to dramatically alter the outcomes of fortune-based contests after the fact.

Also Known As

Fate Points, Luck Points, Force Points, Karma, Hero Points, Possibilities

Related Patterns

Resource

Motivation

The Safety Valve gives players a resource that they can spend to significantly modify the outcome of a contest after the dice have been rolled (or cards drawn, etc.) in a fortune-based contest. The resource is generally very scarce, so players will only spend it when events go far askew from what they would like to see.

Applicability

You should consider the Safety Valve pattern if you want players to have limited veto power over random chance to keep the story flowing in a direction they enjoy (usually for "self-preservation").

There is nothing inherently wrong with adding a Safety Valve to a game, but it may be a strong indicator that your game has a design flaw elsewhere. For example, adding a Safety Valve to help prevent character death introduces a contradiction in your game design. On the one hand, you have rules allowing the possibility of character death. On the other, you have rules preventing it. If you find yourself adding a rule to block the consequences of some other rule, then you may need to re-think that other rule. If you seek to prevent character death, for example, then character death is likely opposed to what you envision your game to be about. If that is the case, you need to seriously consider eliminating character death as a possibility rather than tack on an additional rule to prevent it.

Consequences

The Safety Valve pattern does a decent job at what it sets out to do. It gives players more control over their character's destiny by allowing them to alter rolls that are entirely out-of-whack with their goals. It essentially becomes part of a game's contest rules. However, adding a Safety Valve complicates a system by adding an exception to the normal game flow.

Implementation Concerns

Safety Valves are generally designed for judicious use. The goal is usually accomplished by giving players a resource that is extremely scarce, or alternatively, by

using the resource for more than one purpose, such as character advancement. If a player must decide between spending the resource now for immediate gain or saving it for long-term gain, an inherent conflict is created which can increase tension for the player (if not the character).

If the resource is sufficiently scarce, a player may be able to overturn only a handful of rolls throughout a character's career. Thus, players will naturally want to hoard their Safety Valve resource for moments of extreme urgency. The problem with this is that the scarcity of the resource and the desire to hoard it means that a rule is added to a game, complicates it, and receives inordinate amounts of player attention all for something that rarely has an impact on anything. One way to keep players from hoarding their Safety Valve resource is to refresh the resource periodically and not allow any "left over" resource points to be carried over from one refresh to another. For example, you could give each player one re-roll per scene: they either use it or lose it.

Samples

Let's design a Safety Valve that encourages player use, but which is scarce enough that its use will make a strong statement about what players want to see:

Put "Fate" in the Player's Hands

One time per game session, each player is given the option to influence "Fate" one time. When a die roll goes against the story line that a player would like to see, he can use his Fate and force a re-roll. Note that this rule allows any player to Fate any roll, including those of the Game Master or other player. If the re-rolled result is contrary to what another player desires, that other player may use his Fate (if he has not yet used it) to force yet another re-roll. This may continue until all possible Fates have been used. Unused Fates cannot be carried over from one session to the next. They are either used or lost.

Known Uses

HARP gives characters "Fate Points," which is a scarce resource their players can spend to alter a contest's results significantly. A roll can either be made much more potent or injuries can be greatly reduced. Characters start with 3 Fate Points and more can be bought by spending Development Points, which are used to advance the character's effectiveness. One Fate Point costs 5 Development Points (which is quite expensive).

TORG gives characters a resource called "Possibilities." This resource is used to increase character effectiveness by raising attribute and skill ranks. TORG also allows players to spend Possibility points as a form of Safety Valve. Spending one Possibility allows a player to add a second die roll to an unsatisfactory die roll. If the new d20 roll is less than 10, it counts as a 10. Thus, spending a point of Possibility significantly boosts a character's immediate effectiveness. Possibilities can also be used to immediately reduce damage received by a character.

Warhammer gives characters a “Fate Points” resource that acts as a Safety Valve. Spending a Fate Point allows a character cheat death. For example, it can significantly lessen the effects of critical hits so that a killing blow becomes a minor injury. Or, a character could walk away from a cave-in that would ordinarily have left him crushed to death.

Character Makeup Patterns

Attribute

Intent

Provide a means to gauge universal character aspects within a game environment.

Also Known As

Ability, Stat

Related Patterns

Point-Spend Attribute, Random Attribute, Resource, Template, Trait

Motivation

In order to judge the key capabilities of a given player's character within an imagined world, games often give characters attributes, or stats. Attributes are specific, named gauges usually associated with numbers that have a pre-determined meaning within the game's rule system (although non-numerical attributes are certainly possible as well). In many games, the types of attributes are of a pre-determined fixed number, such as Strength, Dexterity, Intelligence, etc. However, this is not always so. Some games allow attributes to be made up to suit a gaming group's needs. What is important is that attributes

- provide guidelines on how the character can affect the important aspects of the game environment,
- are defined either by the game itself or by a gaming group before play begins,
- are the same for all characters of a given type (they may be different for player and non-player characters).

Attributes allow a player to get a handle on the relative impact a Player Character has on an imagined world works and to what extent he can manipulate the world through his character.

Systems that provide pre-defined attributes can go into great detail in providing rules to appropriately handle game situations. This, in turn, allows the game rules to provide an appropriate level of "crunch," or mechanical detail, in important game situations. Attributes also alleviate some of the need to make spot judgment calls that some players may find unfair.

To focus on the core aspects of a particular genre, some games assign different (but pre-defined) attributes to different characters, depending on the role a particular character plays in a story. As long as all characters portraying a specific role are assigned the same pre-defined aspects, the game could be described as following the Attribute pattern (although it could be argued that it follows the Class pattern instead).

Applicability

Use the Attribute Pattern when you want to provide aspects that are common to all characters where these aspects affect game play in important ways. The majority of role-playing games use this pattern. Some games lack any common character aspects or limits. Childhood games such as "Cops and Robbers" and "Cowboys and Indians" frequently end up with insoluble arguments of "I shot you," "Na-uh," "Did so!"… where the game provides no means to allow the participants to resolve their dispute. If you are looking for a happy medium between these options, you may want to consider the Trait pattern instead.

Consequences

The primary benefit of the Attribute pattern is that it provides a means by which character interaction with the game world can be gauged, thereby reducing the amount of time spent in debates over what a character can or cannot accomplish within a particular game setting. The drawback is that it introduces an artificial mechanism into play that some people may find intrusive. Such people may prefer pure storytelling or freeform style games to more structured games incorporating character attributes.

Implementation Concerns

In using the Attribute pattern, you should consider whether the *number* of attributes your game assigns each character is going to be pre-determined or variable or some combination of the two. If you are going to allow a variable aspect to attribute selection, you need to determine whether you are going to provide a limited list of pre-determined attributes from which to choose or allow the players complete freedom in their choices.

Having a fixed number of attributes or a list from which to choose allows you to tailor those attributes specifically toward game goals and allows you to provide a thorough description of how each attribute affects play, but limits groups in deciding what attributes they find important in their own games.

If a list of pre-defined attributes is used, then the options available to a group in customizing their gaming sessions widen as the list grows. With proper attention to the writing of each attribute description, such a list can provide a great deal of flexibility in campaign design without introducing ambiguity. But, the longer the list becomes, the greater the workload that is placed on the game writers' shoulders. And, even very long lists of pre-defined options can seem confining to some players. Keep in mind, though, that as the game's designer, it is your job to make sure the game's rules focus on its core ideas. What *you* find important in the game is key, because you understand what your game is really about. Creating lists of attributes is fine if that supports your vision. But, if you find yourself creating ever-longer lists of potential attributes you might want to step back and re-assess your approach.

Having a variable number of unspecified attributes where each group dreams up its own attribute list allows for even more flexibility. However, this flexibility introduces difficulties in deciding what exactly each custom attribute really means and how they

interact in conflicts. Such a system must pay particular attention to the descriptions they provide of what constitutes a valid attribute and how they affect one another in game play. While this approach may generalize your system to being usable by a wider audience, it also dilutes your game's focus and puts a great deal of game design burden on the players' shoulders. This may not be a good idea, because game design is hard to get right. Current thinking in game design is that tightly focused, well-designed games are more consistently fun than games lacking focus. If your game ends up being a drag to play, people simply will not play it regardless of how universal its rules.

One way to gain flexibility and yet maintain a focused design is to design your game around a fixed number of unspecified attributes. That is, set up the mechanics of how specific character attributes fit into the game system, but leave the actual interpretation of those attributes open. You could, for example, give each character an "Obsession" attribute whose mechanical interrelationships with other gauges is fully specified and yet allow each player to customize its meaning for his character. So, one character could have an Obsession for cars, which allows that attribute to apply whenever that character was involved in car scenes. Another might have an Obsession for horses, so his attribute would apply in scenes dealing with equines.

In a game using the Attribute pattern, you should give some thought to the attributes that are assigned to non-player characters and objects as well, such as doors, rocks, griffons, robots, and/or trees. To keep a game as concise as possible, it is reasonable to put some effort into designing an attribute set that can be used universally throughout the game environment. That is, play may run more smoothly if all interacting characters, both player and non-player, use the same collection of stats. Of course, this is not true for all games. You will have to decide based on your design goals.

Samples

Comparing the attributes of one character directly to those of another may provide a yes/no answer directly in your game. For an example of this style of attribute usage, Bill the Barbarian's Strength rating is a 5 while Carla the Crusader's Weight is a 3. Therefore, Bill can lift Carla because his Strength is greater than her Weight.

In other games, attributes modify die rolls in determining success. As a simple example, Larry the Lush begs a few dollars from a passing stranger. He hasn't eaten since yesterday at noon, but he is standing outside a liquor store. Larry's Alcoholism rating is a 3 while his Hunger rating is a 4. Larry's player rolls two d6. To the result of one he adds his Alcoholism score and to the other he adds his Hunger rating. The former sums to a 7 while the latter adds up to a 5. Larry's addiction gets the best of him and he stumbles into the shop to buy a bottle of cheap booze in which to drown his hunger.

Known Uses

Fudge allows a Game Master to determine the set of common "attributes" all characters share in his game world. Words are used to describe attributes as Terrible, Poor, Mediocre, Fair, Good, Great, and Superb. These are rated according to the "scale" of the game, where "Fair" is the norm. In a game where the characters are all snails, the average snail speed would be "Fair." Similarly, in a game where the characters are centaurs, the average centaur speed would be "Fair."

Rolemaster Fantasy Role Playing has 10 "stats" of Agility, Constitution, Memory, Reasoning, Self Discipline, Empathy, Intuition, Presence, Strength, and Quickness. Players are given a resource pool of 600+10d10 points to distribute among these stats with certain limitations based on the character's class. Optionally, a player may distribute 660 points rather than take his chances on rolling dice. Stat values above 90 become exponentially more expensive.

Class

Intent

Provide a means to quickly assign a group of abilities to a character and, at the same time, protect character niches to ensure each character plays a meaningful role within the game.

Also Known As

Profession, Occupation

Related Patterns

Class Tree, Skill, Template

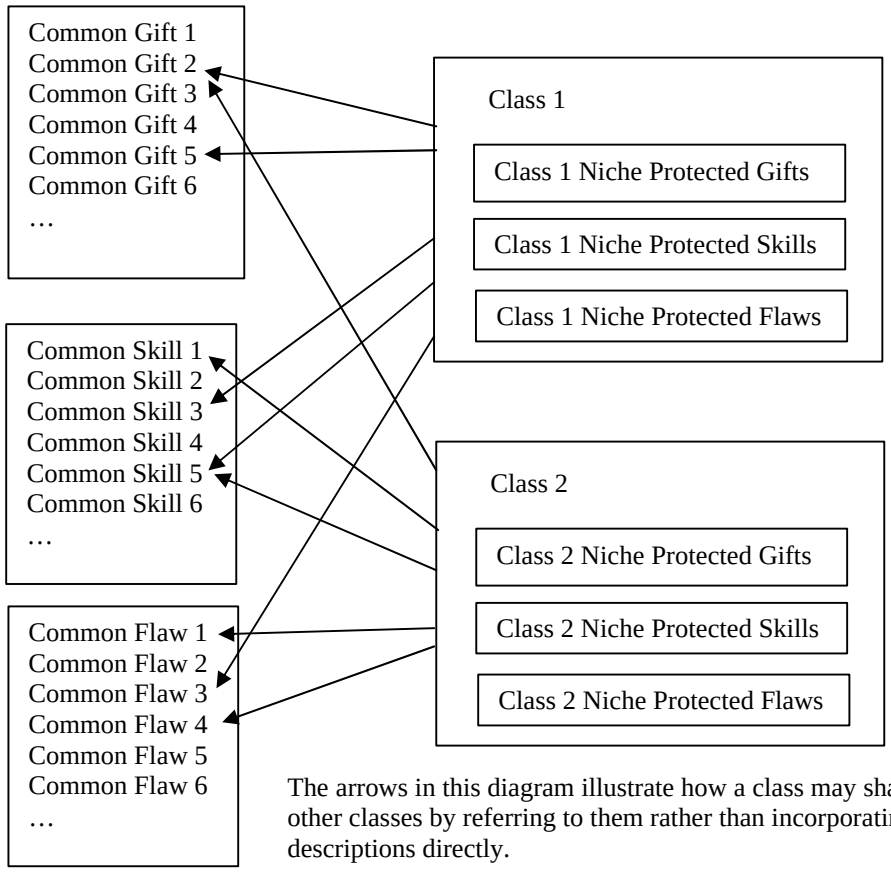
Motivation

A class is a collection of flaws, gifts, skills, and/or handicaps that are given to a character when his player elects the class. In the "pure" form presented in this pattern, a player is allowed to choose only a very few classes for his character, perhaps only one. After this choice, his character is not allowed to "wander" outside the confines of what his class choices allow. In some games, classes are applied only during character generation, but this is not always the case.

The class pattern accomplishes a number of important goals. First, it partitions off a subset of the rules so that beginning players only have to look at the rules directly pertinent to the characters they are playing. Second, it greatly reduces the number of decisions that have to be made by a player in creating a character, thus making character generation faster. Primarily, though, it partitions characters into specific "professions" so that character abilities overlap those of others to only a minimal degree. This helps ensure that each character has a useful, unique role to play within the game environment.

Some games "soften" the Class pattern to increase flexibility at the cost of diminished niche protection. For example, a game using the Rank pattern could use classes as a means to specify what skill ranks a character may gain "cheaply." In a game with this design, a player can have his character wander outside his class "boundaries," but only by suffering some additional cost or penalty.

Example Structure



The arrows in this diagram illustrate how a class may share abilities with other classes by referring to them rather than incorporating their descriptions directly.

Applicability

Games whose primary characters tend to have many similar aspects are more suited to the Class pattern than games whose characters naturally vary widely. This is because games having characters with a large number of commonalities need to put special emphasis (and niche protection) on the characteristics that distinguish one character type from another. For example, in a game where the central characters are all New York City cops, all characters are likely to have some abilities in common (namely, those abilities normally taught at Police Academies such as firing handguns, unarmed combat, etc.). So, you might want to clearly distinguish between the special categories of Detectives, Beat Cops, SWAT officers, and Forensics experts. On the other hand, games allowing characters with radically different abilities need not be so concerned with niche protection, since the players themselves are better able to define their own unique roles.

Another area where the class pattern is appropriate is in games where you want to reduce the number of player choices. For example, in designing games for young children, you may decide to keep the number of player decisions to an absolute minimum. Here, you might be best served by the basic Class pattern.

Use the Class pattern when you want to

1) minimize the number of decisions that players need to make when generating their characters,
2) allow players to learn only the subset of rules pertinent to their characters,
3) protect character niches so that characters with different classes play different, meaningful roles within the game.

Note that if you are primarily interested in reducing the amount of bookwork needed to initially generate a character, the Template pattern satisfies that goal. If niche protection is unimportant to your game concept, you might want to consider that pattern instead.

Consequences

The Class pattern can be extremely confining to some seasoned gamers without additional flexibility built into the system. This is understandable, because one of the main reasons gamers play role-playing games is for the freedom these kinds of games promise. The basic Class pattern purposefully reduces character flexibility to attain benefits that might be better achieved in other ways. So, careful consideration of alternatives should be made before making the decision to use this pattern in its "pure" form. For example, a great deal of flexibility can be gained in a class-based system through the use of the Class Tree pattern.

Implementation Concerns

Since classes generally provide an overall benefit to the character, some means of limiting the number of classes a given character may obtain must be provided (unless a game's classes provide sufficient drawbacks to balance their benefits).

One of the primary motivations for using the Class pattern is to clearly delineate character niches. So, it is important to avoid making other design choices that might interfere with this partitioning. For example, in a fantasy game, granting wizards the ability to heal would trespass on what is often considered to be the domain of priests. Simplicity is another key reason for selecting the Class pattern, so it makes sense to remain consistent with that goal and keep the classes themselves from requiring players to make very many additional decisions. Once again, if you find it hard to stick to this principle in practice, you may want to consider using some other pattern instead.

Samples

The following might be an example of a class in a game containing medieval thieves:

Bandit

Bandits often group together to form ambushes on caravans and wealthy nobility. They constantly try to invent new ways to trap and overcome opponents normally considered too powerful to defeat. Of course, they frequently set up their surprises in ravines and mountain passes, but imaginative

ploys always inspire these thieves. They realize that only a limited number of ambushes are safe at a given spot before some *real* force shows up.

Skills: Stealth, Setting Traps, Climbing Walls, Horsemanship, Tracking
Weapon Proficiencies: Wielding medium hand held weapons, Firing crossbows, Using large entrapment weapons

Known Uses

RIFTS has 22 "Occupational Character Classes," including Borgs, Headhunters, Vagabonds, Techno-Wizards, and the like. Each class has attribute requirements (see the Attribute pattern) that must be met before selecting the class. Once selected, the class bestows a specific list of fixed skills on the character. The classes also provide lists of additional skills from which the player is expected to choose a specified number at various "levels" (see the Rank pattern).

Rolemaster Fantasy Role Playing has 9 "Professions," including Fighter, Thief, Magician, Cleric, and others. These classes each have "Prime Stats" requirements that specify minimum attribute values that must be met to gain the class. Each class provides spell lists (for spell casters), bonuses of various sorts (such as for armor and weapon use), costs for raising various skills (see the Rank pattern), and a number of "Training Packages" (each of which is essentially a group of skills) along with how much each package costs should the player elect it for his character.

Class Tree

Intent

Provide the niche-protection benefits of the Class pattern while allowing players flexibility in customizing their characters along with the ability to gain new abilities as game play progresses.

Also Known As

Class Path, Career Path, Life Path

Related Patterns

Class, Loose Coupling, Template, Traits

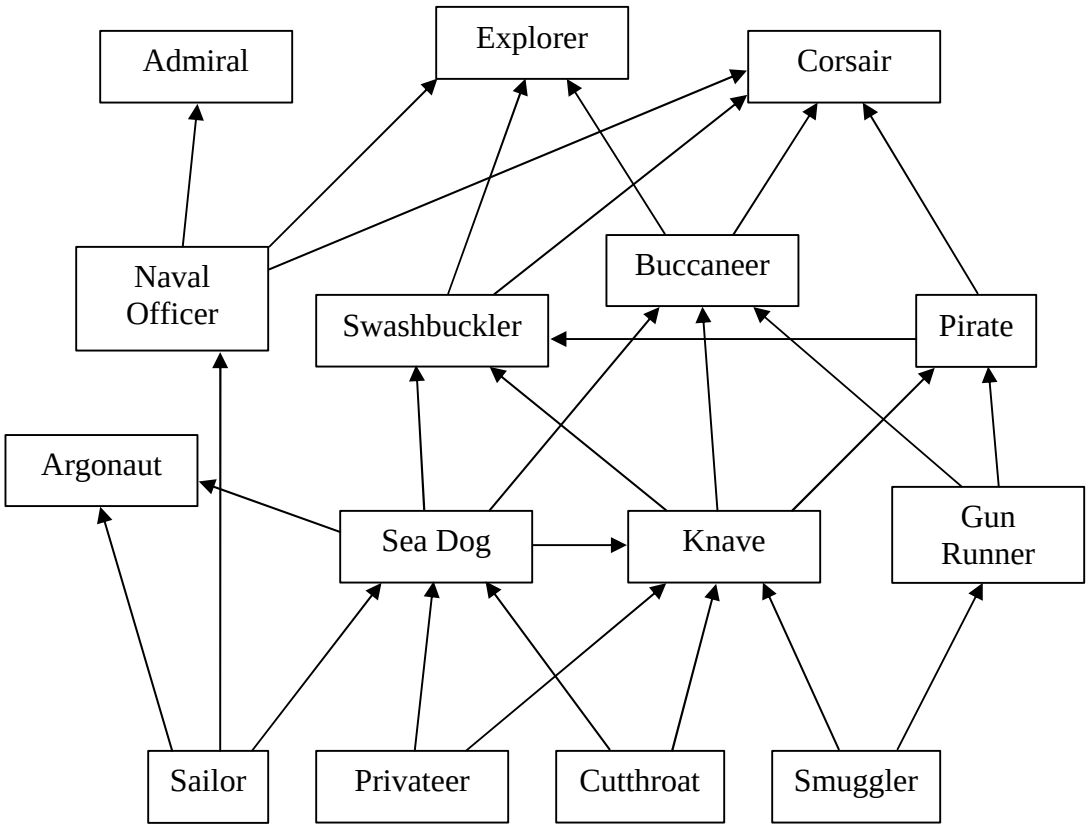
Motivation

The Class Tree pattern is closely related to the Class pattern and is based on the fundamental concept of classes, but adds flexibility by allowing a player to select multiple classes at different points in his character's career. The basic class pattern allows a character to gain more than one class in some games, so in this regard the class tree is not unique. What distinguishes the Class Tree pattern from the basic Class pattern is that Class Trees allow career options to increase as play progresses. That is, some classes are unavailable to a character until certain conditions are met. How those

conditions are satisfied varies from game to game, but a class that provides more benefit than another will generally be more difficult to attain. Since character abilities tend to increase as more classes are gained, the more classes a character has, the greater the number of classes for which he qualifies. If a character can only obtain a limited number of classes, though, the player is forced to make important decisions concerning his character's career.

When a new class is obtained, it expands the character's in-game options through increased skill sets. As he gains each new class, the character changes qualitatively. This can breathe "new life" into a character as the player has new potential activities to explore. It also has the benefit of requiring the player to make a statement of what capabilities are important to his character, because the classes chosen must be made at the expense of other career options. Since the options branch and increase in number over time, advancement opportunities can be thought of as a tree structure.

Example Structure



The arrows in this diagram illustrate potential advancement opportunities for any character possessing a given class.

Applicability

The Class Tree pattern can be used in games when you want to

1) Simplify character generation by reducing the number of decisions that need to be made by players,
2) Increase flexibility in character design over a basic class system,
3) Provide niche protection for different "types" of characters,
4) Allow expansion of character skill sets as play progresses.

If niche protection is unimportant to your game concept, you might want to consider using the Template or Traits patterns instead.

Consequences

The Class Tree pattern puts a lot of work on the game designer's shoulders, because it requires him to write up detailed descriptions of a fairly large number of interesting classes pertinent to game play.

Because of the potentially bewildering array of class options demanded by the Class Tree pattern, character generation for beginning players can become more complex if the players must assess their preferred choices from all those available. This conflicts with the basic Class pattern's goal of making it easier for beginning players to get characters up and running quickly. For this reason it is best to keep the number of "starting" classes to a fairly low number and partition them from the other class options for easy perusal. Another option is to make the choice of class random, so beginning players do not need to make informed decisions. They just roll dice and take what they get. Gamers wanting a great deal of control over their character's makeup are likely to find this solution to be unsatisfying, however.

When implemented properly, the Class Tree pattern becomes part of the game's reward system. Players will tend to focus their characters on attaining particular "advanced" classes that interest them and often get a thrill when their characters finally gain them.

As play progresses in games using the Class Tree pattern, the characters' list of in-game options expands as the number of abilities bestowed by their growing class list increases. Because of this gradual but punctuated enhancement to character effectiveness, players tend to find renewed interest in their characters as new play options become available.

Implementation Concerns

The Class Tree pattern has many of the same implementation concerns as that of the basic Class pattern. You will most likely need to have some limitation on the number of classes that a character can select, so that no character is capable of gaining all classes and thereby negate the niche protection features of the pattern. One way to do

this is to have a resource which players can use to buy their classes at some pre-defined "cost" (see the Resource pattern).

The niche protection characteristic of the Class Tree pattern is one of its most important features, so care must be taken to ensure that the skills of one class do not trespass too far into the realms of other classes. This can be difficult at times, because the sheer number of classes that must be written to make the Class Tree pattern sufficiently flexible make some overlap between classes virtually inevitable. To mitigate this problem, you can have different classes that overlap in their skills make-up have other important differences. If the Skills Rank pattern is being used, one way to do this would be to make the overlapping skills have significant differences in the cost needed to gain ranks, even though the skills themselves are similar.

Since a class tree necessitates some kind of interaction between classes and must somehow allow advancement from one class to another, some means of identifying what classes are available to a character at any given time is needed. If care is not taken concerning how these options are presented, problems in future game expansion can arise. For example, if you decide to write and release a supplement of new class options after the game's core rulebook is released, you may encounter difficulties in allowing characters to take advantage of the new classes since the core rulebook has no knowledge of their existence. For information on how these problems can be mitigated, see the Loose Coupling pattern.

Samples

The following might be an example of a class in a game containing Caribbean-style pirates:

Swashbuckler

These flamboyant swordsmen travel far and wide in search of adventure. Many stories tell of swashbucklers swinging on chandeliers to quickly rescue maidens. They relate how these characters leap from ship to ship to combat rivals with swordplay. Although many swashbucklers constantly break the laws of established governments, all seek merely to have a good time. Jovial to the end, swashbucklers often swill rum, mead, or wine when engaged in combat and laugh heartily at their own mistakes in battle.

Attribute Requirements
Agility 2, Perception 2

Prerequisites
The character must be able to swim (via the Swimming gift) and must have attained 8th rank in wielding Sabre, Cutlass, or Foil.

Gifts
Acrobatics, Dancing, Etiquette, Holding Liquor, Speaking Foreign Languages

Skills
Climbing Walls (+1), Disarming Opponents (+3), Florentine (+3), Inspiring Loyalty (+2), Navigating (+2), Raising Morale (+2), Seamanship (+1), Weapons Resourcefulness (+3)

Weapon Proficiencies
Wielding Sabre (+2), Cutlass (+2), and Foil (+2)

Known Uses

Dungeons & Dragons v.3.5 has 11 basic classes, including Barbarian, Bard, Cleric, Druid, Fighter, Monk, Paladin, Ranger, Rogue, Sorcerer, and Wizard. After a character has advanced sufficiently in "levels" (see the Level pattern), he has the option of obtaining any of a number of "Prestige Classes." The "Dungeon Master's Guide" lists only 16 prestige classes including Archmage, Blackguard, and Hierophant, but there are numerous D&D supplements that provide a host of others. Each prestige class provides a list of "Requirements" needed to attain the class. The basic rules allow characters to progress based on the total number of levels of all classes obtained. A 10^{th} level fighter/7^{th} level wizard is treated as a 17^{th} level character overall. So, he would not progress to his next level until earning the experience necessary for an 18^{th} level character. So, the system discourages having any character branch out into too many classes.

Warhammer Fantasy Role Play has 4 broad class categories of Warriors, Rangers, Rogues, and Academics. Once a player selects one of these categories, he rolls randomly to determine which of the "Basic Careers" associated with that broad class that his character starts with. The game has 63 "Basic Careers" and 36 "Advanced Careers," most of which have "Career Exits" that specify which careers are available as advancement opportunities to a character that currently has that career. When a player decides to have his character take a career exit, he must spend an appropriate amount of experience points to do so. When he does, his previous career is over and his new potential "Career Exits" are those associated with his new career. A new career gains him new options but loses old ones. (Perhaps this would be better described as a "Class Vine" rather than a "Class Tree.") Each career provides a list of skills and pre-defined limits on how far a character can progress in his abilities while practicing the career.

Gift

Intent

Provide a means for players to customize their characters with specialized, well-documented abilities that do not improve in effectiveness as play progresses.

Also Known As

Feats, Talents

Related Patterns

Class, Skill, Trait

Motivation

Most role-playing games allow players to customize their characters with new abilities. Sometimes, it makes sense to allow those abilities to improve over time. In other cases, it makes more sense to simply bestow an ability upon a character as a well-defined "gift" and forego any possible advancement in it. There are a few reasons why this might be done.

Some gifts allow a game designer to explicitly handle situations that are likely to arise, but are not a central focus of the game. For example, a game may be set in a world where literacy is far from universal. However, the game designer does not want to require players to deal with the bookkeeping work of a full-blown "Reading" skill. In his viewpoint, knowing whether a character can read is important because it may have social implications, but trying to deal with the minutiae of what 1st vs. 5th rank in the skill of "Reading" means is beyond the level of detail needed by the game. On the other hand, a game set on a university campus may have need for that level of detail. (In such a case, the Rank pattern may be more appropriate.)

In other cases, it just doesn't make conceptual sense for an ability to increase as time passes. For example, if the ability of "night vision" is bestowed on a character for choosing a cave-dwelling race, then the game designer may decide that there is no justification for a character to improve in that area after the gift is bestowed.

In still other cases, a gift might be used as a sort of "toggle switch" to turn on other, related abilities allowed by a game. For example, a character may have the skill "Armor Smithing," which allows him to craft ring armor at 3rd rank, mail armor at 5th rank, plate armor at 8th rank, etc. However, he cannot craft *magical* armor or the corresponding type unless he obtains the gift of "Crafting Magical Armor." Presumably, going through the effort and cost of obtaining this gift would open up a whole realm of armor crafting possibilities that a character otherwise would not have.

Applicability

Use the Gift pattern when you want to

1) Allow players to differentiate the abilities of their characters from others in their group,
2) Keep those abilities from improving as play progresses,
3) Provide detailed descriptions of what each of those abilities accomplishes.

Goal 1 can be satisfied by the Trait pattern. It would be trivial to modify the Trait pattern to keep some traits from improving as play progresses as well, thus satisfying goal 2. So, if detailed write-ups of character "gifts" are something you want to avoid, you should consider using the Trait pattern instead.

If you would prefer an ability to increase in effectiveness as play continues, you should consider using the Level, Skill, or Rank patterns instead.

Consequences

The Gift pattern is often used in conjunction with the Skills pattern. Like the Skills pattern, Gifts place a significant burden on a game writer's shoulders, in that they require detailed write-ups of the various abilities presented as options. Gifts are one way in which a "rules-light" game becomes "rules-heavy" (which is not necessarily a bad thing). On the other hand, gifts provide a powerful way for a game designer to direct play toward the kinds of scenarios he envisions his game to be about. If well written, gift descriptions can answer most questions that will arise in game play concerning the ability, thus avoiding debate about trivial details during play. All of this costs a certain amount of flexibility, though. Some players find skills and gifts based games to be confining regardless of how many gifts are offered as options.

Implementation Concerns

You will need to decide when, how many, and what kinds of gifts a character can gain. One possible way would be to create some kind of resource that a player can "spend" to purchase gifts (see the Resource pattern). Another might be that gifts are bestowed by race, one of which a player must choose for his character initially. A third option is that gifts are bestowed by classes (see the Class and Class Tree patterns) which are themselves limited by some means.

In writing up your gift descriptions, you will need to be careful to temper your enthusiasm for creating "kewl powers." Gifts can be used effectively to augment a game, but can just as easily upset game balance if you do not maintain some amount of consistency from one description to the next. One way to allow gifts of varying effectiveness without upsetting the integrity of your game is to have the more effective gifts cost more (see the Resource pattern).

Samples

A game might specify a "Ciphering" gift in the following way:

> **Ciphering**
> Gift Cost: 1 Character Point
> Ciphering enables a character to perform simple arithmetic. He understands the basic concepts of numbers, tables, and charts. Of course, almost any character can keep track of numbers under 20. Nevertheless, a character with this talent need not stop after he runs out of fingers and toes. He easily deals with addition and subtraction and can multiply and divide with difficulty.

Known Uses

Dungeons & Dragons v.3.5 has a list of 109 "Feats" such as "Brew Potion," "Dodge," and "Quick Draw." Feats are bestowed both at the time the character is initially created and as the character advances in levels (see the Level pattern).

HERO System 5th Edition has a list of 18 "Talents" such as "Ambidexterity" and "Eidetic Memory." Talents are purchased through the expenditure of "Character Points," which is a resource spent by players in designing their characters.

Warhammer Fantasy Role Play has a list of 133 "skills" such as "Gamble," "Pick Lock," and "Spot Traps." These "skills" actually match the Gifts pattern because they do not change in effectiveness as play progresses. These gifts are bestowed upon characters as they gain new "professions" (see the Class pattern) throughout their careers.

Hit Points

Intent

Provide a resource to gauge when a character is incapacitated or dies.

Also Known As

Damage Capacity, Health, Life, Vitality

Related Patterns

Endgame, Resource, Trauma Gauge, Wound Trait

Motivation

Role-playing games originally evolved from various war games. As such, many of them deal with combat and death. Such games require some means to determine when a character becomes incapacitated or dies due to injuries. Hit Points are among the earliest techniques devised to accomplish this goal. If a game specifies a resource from which "damage" is subtracted that causes a character to be rendered helpless or dead at pre-specified points, it follows the Hit Points pattern. So, the Hit Points pattern is really just a specialization of the Resource pattern.

A pure Hit Points gauge ignores the realistic incremental effects of various wounds in favor of simplicity. It also avoids the "death spiral" effects that can arise from systems that take these incremental effects into account (see the Trauma Gauge and Wound Trait patterns).

It is very common for games using Hit Points to state that a character is rendered incapacitated when damage lowers the characters Hit Points to 0 and death results when some pre-set negative number is reached. Other games specify that characters die at 0 Hit Points. Some Hit Point based games provide penalties to characters whose Hit Points fall to very low numbers in order to simulate the debilitating effects of pain and injury. (This is really a blending of the Hit Points and Trauma Gauge patterns.)

Applicability

The Hit Points pattern is appropriate in games where the designer

1) Wants to incapacitate characters when the incremental effects of injury build up to critical levels,
2) Is willing to either accept the "unrealistically" sudden loss of effectiveness the pattern implements or provide other means to simulate incremental wound effects.

If you want to simulate the gradual loss of combat proficiency as a character accrues damage, you might want to consider the Trauma Gauge or Wound Trait patterns as alternatives or as supplements to the Hit Points pattern. If you want to guarantee the

survival of characters until their roles are fully played out in a storyline, you should consider the Endgame pattern as an alternative.

On the other hand, the nature of your game may have nothing to do with death. If so, you probably want to avoid Hit Points or any of its related patterns altogether. For example, if you are designing a role-playing game where all of the characters are cartoon characters, it would be a logical design goal to ensure that no character can possibly die. You might decide that the worst that can happen to a cartoon character is for it to become a broken and mangled ruin at the bottom of a very deep canyon with the shadow of a falling boulder growing around it. In a subsequent scene, the character might be wrapped in bandages and using crutches, or he might be perfectly healthy based purely on what the players find to be most amusing. Ensuring the impossibility of death can be accomplished by avoiding the introduction of any rules into your game describing death as a possibility. You might even want to mention somewhere that this omission was intentional for players who are oriented toward war gaming.

Consequences

The primary benefit of the Hit Points pattern is that it provides a simple way for players to track and manage a resource specifying how "close" their character is to death. In its pure form, Hit Points impose no penalties on character actions as damage is accrued. Consequently, characters are as effective when severely damaged as they are when completely healthy up until the critical state where the character suddenly loses all capability. For some players, this aspect of the pattern is highly unrealistic, causing their "suspension of disbelief" to be challenged. Others are willing to ignore this characteristic in favor of the simplicity and prolonged character effectiveness it affords.

Implementation Concerns

If you use the Hit Points pattern in your game, you have several factors to consider:

1) How are a character's Hit Points determined?
2) How is "damage" calculated and accrued?
3) How are lost Hit Points regained?

In many games, Hit Points are generated by having the players roll dice. The number of dice rolled is usually tied to the power or "experience" of the character. Most often, games using this technique also use the Level pattern and have the player roll one (or more) die per character "level." Sometimes, the size of the die used (d4, d6, d8, etc.) depends on the character's class or race (see the Class pattern). This "randomized" technique has a number of problems of which you should be aware.

First, newly generated characters usually start out with a very small number of Hit Points, possibly only 1. In games employing a game master, the survivability of new characters largely becomes a matter of how skilled the game master is at keeping characters alive, since a blow that would be considered a very light wound to more powerful characters can easily slay a new character. Games that do not employ a game

master would not even have GM fiat as a buffer, and so new characters would naturally suffer from an even higher fatality rate.

Second, the range of possible Hit Point values expands as the game progresses. Consequently, gearing obstacles to a specific power level becomes more and more difficult as power levels increase. The best that can be done is to target the "average" Hit Points value for a given power level. And, if the size of dice used varies from character to character based on race or class constraints, deciding on what "average" means becomes problematic in itself.

Third, Hit Points are generally a crucial characteristic for in-game success. So, groups will often come up with "house rules" to deal with situations where low Hit Point numbers are rolled. For example, a player that rolls a 1 on his Hit Point die will be allowed to re-roll rather than take the minimum. At the very least, these kinds of house rules highlight the weaknesses of allowing pure chance to determine so crucial a characteristic. At worst, they can disrupt your carefully crafted game design by shifting the meaning of "average" in some unanticipated way.

Finally, continually escalating Hit Point values forces the game design into an ever increasing arms race, so that the game challenges powerful characters as well as beginning characters. It is easy to fall into the trap of providing characters with more potent versions of similar abilities to those already possessed. Doing this merely updates character abilities to deal with the ballooning Hit Point values, rather than giving players truly new ways in which to explore their game environment.

All of this is not to say that the Hit Point pattern is invalid for your game design, only that leaving the Hit Point determination to random chance is questionable. All of the aforementioned problems are the direct result of dice rolls and rapidly swelling Hit Point values. The Hit Points pattern requires neither of these characteristics. For example, Hit Points could be "bought" through the expenditure of some resource in a manner similar to what is described in the Point-Spend Attributes pattern. That way, players would be free to set their own Hit Point values, which would be determined from their own ambitions after weighing the importance of Hit Points against other concerns. Alternately, you could have Hit Points be derived from other attributes via some formula that would ensure beginning characters have a reasonable chance at surviving. If the attributes from which Hit Points are derived are Point-Spend Attributes, then players would again have indirect but total control over their characters' Hit Point values.

At the same time you are figuring out how Hit Points are calculated, you need to be thinking about how you want them to be lost. Hit Points are fundamentally a resource, which means they can be "spent" in some fashion. Generally, Hit Points are "gambled" by having characters participate in combat where damage can be sustained in various ways. This means that you must also decide how damage is calculated. Some games use dice rolls to calculate the damage values of a weapon strike. For example, a short sword might deliver 1d8 damage whenever it hits a foe. Randomizing weapon damage

is far less problematic than randomizing Hit Points, assuming that the upper damage range is unlikely to kill a healthy character, because the effects are presumably temporary in that characters can usually "heal" through rest or medical attention (magical or otherwise depending on genre). Even so, rolling extra dice for damage in a "Roll to Hit / Roll to Damage" fashion is often unnecessary. Some games combine both rolls into a single roll where the degree of success of a weapon strike determines the amount of damage delivered. That doesn't mean that you shouldn't design your game with a separate damage roll, only that you should do so based on your goals and not on some pre-conceived notion that they must be separate actions.

Finally, you need to consider how characters regain lost Hit Points. A story can be brought to a grinding halt because players with badly injured characters don't want to foolishly risk their lives. So, lacking any viable alternative, the players will simply have their characters "rest," however dull and dreary that activity proves to be. (This assumes, of course, that "resting" has beneficial effects on Hit Points.) If your game genre allows for it, you can give characters some supernatural means of regaining Hit Points. For example, in a game where all characters are ghouls, you could specify that eating raw flesh invigorates the characters and heals wounds. In a game where characters are werewolves, stating that werewolves regenerate at a rapid rate is reasonable. You must strike a balance between allowing players to quickly resume interesting play after having suffered serious injury and defeating the entire purpose of having Hit Points in the first place. If you give characters a supply of Hit Points that refreshes as quickly as they are lost, then characters will never be in danger of incapacitation or death, which is the whole reason for tracking Hit Points. Of course, games where characters are normal humans have no easy excuse for fast recovery from near-mortal wounds. Even so, artificial means such as magical or "high tech" healing can be used to speed things along.

Note that regaining Hit Points need not be fast in "game" time, only that it should happen quickly in "real" time. Suppose a mountain man character in an Old American West style game is badly mauled while battling a bear and drags himself to his cabin to recuperate. Such a game cannot easily incorporate magical or high tech excuses for speedy recovery and remain in-genre. But, it can encourage the rapid passing of time to maintain dramatic tension. "Okay, three months later your character has a bad scar on his left side but is otherwise none the worse for wear."

Samples

In a game having the attributes of Strength and Constitution (see the Attribute pattern), Hit Points could be determined with a formula such as the following:

$$\text{Hit Points} = 50 + 5 \times (\text{Strength} + \text{Constitution}).$$

Known Uses

Dungeons & Dragons v3.5 determines "Hit Points" randomly. Whenever a character gains a level (see the Level pattern), its player rolls a die, modifies the result by an

adjustment based on the character's Constitution, and adds the result to the character's Hit Point total. The character's class (see the Class pattern) determines the size of die rolled (d4, d6, d8, or d10). Weapon damage is calculated by dice rolls and adding adjustments for magic and strength. Hit Points are regained through rest or magical healing.

The Riddle of Steel relies mainly on the Wound Trait pattern for the effects of damage, but it does have an attribute called "Health" that fits the Hit Point pattern for Blood Loss. Wound Traits in The Riddle of Steel sometimes inflict blood loss. Every turn, a conflict resolution roll is made between a wounded character's current blood loss rating and his Endurance attribute. If the roll fails (i.e., "Blood Loss" wins), the character loses a point of Health. If his Health falls to 0, he dies. Note that Health is an attribute that can be directly used in conflict resolution rolls, but it does not reduce the effectiveness of other attributes, so it doesn't follow the Trauma Gauge pattern. Blood Loss returns at a rate of one point per day of rest unless magical means are employed to speed recovery.

Rolemaster Fantasy Role Playing sets "Hits" equal to the character's bonus in the skill of "Body Development" (see the Skill and Rank patterns). Weapon damage is determined by rolling percentile dice and looking up the results on one of various tables, taking into account both weapon and armor types. Hit Points are regained through rest or magical healing. Note that in Rolemaster, it is actually very difficult to die from hit point loss, since a character's hit points can go down to a large negative value before death results. Consequently, most character death is delivered through Wound Traits ("Crits").

TORG distinguishes between "Shock Damage," "Knockout Conditions," and "Wound Levels." When Shock Damage exceeds a character's "Toughness" attribute, he falls unconscious. Characters recover Shock Damage at a fast rate (one point per minute), but Wounds take a long time to heal. "Wound Level" is effectively a separate Hit Point resource having four levels (actually five, if you count "Unwounded"). These are: Wounded, Heavily Wounded, Mortally Wounded, and Dead. TORG's Knockout conditions are unrelated to Hit Points.

Level

Intent

Provide a rough gauge of a character's survivability in a given environment and, optionally, a number from which skill effectiveness is derived.

Also Known As

Not Applicable

Related Patterns

Class, Rank

Motivation

Any game that provides a number (or small set of numbers) to represent a character's overall level of "experience" or "power" follows the Level pattern. Of course, this usually implies that a character's power increases in some way as play progresses. At the very least, the use of the Level pattern suggests that a character's overall effectiveness can differ from that of other characters in some way.

There are essentially two reasons a game designer might decide to use the Level pattern:

1) To reduce the number of values a player must maintain for his character, and thereby simplify the game design,
2) To provide a quick estimate of a character's chances of overcoming some foe or of successfully completing an adventure.

Adding a "Level" to a character's statistics reduces the amount of bookkeeping a player must maintain if a significant portion of a character's skills have their effectiveness determined by his level. That way, numbers for individual abilities do not have to be maintained as is required by other skill rating patterns (see the Rank pattern). Many players would argue that the cost to flexibility is too high, but the truth of the matter depends entirely on your design goals. Games geared toward young children must be particularly sensitive to complexity issues and you may decide you are justified in applying a Level strategy in such a situation.

Having a simple way to measure a character's overall survivability within a game scenario is also a valid design goal. If you want to write game supplements that provide pre-canned adventures, or "modules," having a concise means of conveying the survivability requirements to the players can prove convenient. Players with "1st level" characters could easily avoid the near-certain disaster of tackling a module clearly labeled "For 4 to 6 characters of 12th to 14th level."

Applicability

Use the Level pattern when your design goals include

1) A need to allow characters to increase in power as play progresses.
2) A need to quickly gauge a character's chances of survival in a game environment where foes can vary widely in their own power and capabilities.
3) Reducing the amount of bookkeeping needed to maintain the effectiveness of a wide variety of abilities. (optional)

If you want to keep your game rules adaptable to computer video games, the Level pattern is suitable as well. Many "role-playing" computer games, both online and offline, use levels as a means to determine what aspects of the game a character can explore. For example, some games allow characters to "group" and adventure only with other characters of a similar level.

On the other hand, a character's chances of survivability are equally easy to gauge in games that have no concept of character "advancement." In such games, all characters have an equal chance of survivability and rewards are provided by means other than the accumulation of power.

Consequences

If a character's effectiveness is directly based on his Level, the pattern can have a big impact on the game. Such a system is necessarily constrained in how much control players have over their individual character abilities. Some RPG veterans refuse to play "Level-based" games due to this built-in inflexibility. But, the pattern does lower the bookkeeping overhead of maintaining a character, so games geared toward very young players can benefit from its use.

Games that use the Level pattern as a "power gauge" and yet still allow abilities to be individually customized sacrifice little or no flexibility by using this pattern. Of course, they also forego any bookkeeping benefits the pattern offers, so the justification of using the pattern is similarly reduced.

Implementation Concerns

When adopting the Level pattern, you should first cogitate on how pervasive its use will be throughout your game. If you want to use it primarily as a power gauge, then you should look at the Rank pattern as an alternate way to determine the potency of character abilities.

If you want to base *some* abilities on Level but use some different measure for others, then you need to construct sharp boundaries between various groups of abilities to avoid confusion. For example, you could decide that all combat abilities should be based on level while all other "skills" are individually ranked. Before making these kinds of distinctions, though, make sure they satisfy some important game goal or you will complicate your design unnecessarily.

You also need to figure out how levels are raised or lowered. Will the level automatically rise by one for every successfully completed adventure (see the Success Reward pattern)? Will you base it off of some notion of character experience? Will it vary based on some subjective judgment of "good" role-play (see the Idiom pattern)?

Samples

A game using the Experience Point pattern may derive a character's Level from the amount of experience points he has accumulated. In such a game, you might see a Level description like the following:

> **Level**
> Your character's Level gives you a quick summation of your character's success to date. It is determined by the number of Experience Points (X.P.) your character has earned according to the following table:

Level	X.P.	Level	X.P.
1	0	9	3200
2	25	10	6400
3	50	11	12800
4	100	12	25600
5	200	13	51200
6	400	14	102400
7	800	15	204800
8	1600	16	409600

Known Uses

Dungeons & Dragons v.3.5 uses levels in conjunction with classes (see the Class pattern). So, a character gains levels within a specific class, such as Fighter or Wizard, as he earns experience points. Many character abilities are based directly off of the character's level, such as combat proficiency and spell casting effectiveness. But, individual skill ranks can be purchased with "skill points" that are earned as characters gain levels (see the Skill and Rank patterns). Characters are limited to a total of 20 levels in all classes. However, supplements can be purchased that provide rules that allow players to continue adventuring above level 20 with "epic" level characters.

HARP bases its levels off of experience points as well. Gaining a level earns a character "development points" from which attribute bonuses, talents, and skill ranks can be purchased (see the Point-Spend Attributes, Gift, Skill, and Rank patterns). Level has no other obvious effect in the game mechanics.

RIFTS follows the pack and also bases its levels off of experience points. Levels are gained at different rates based on experience point tables provided for each character class (see the Class pattern).

Point Spend Attributes

Intent

Provide a means for players to design their characters' attribute values by trading them off against one another.

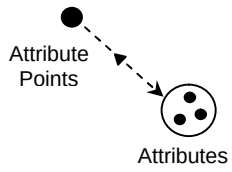
Also Known As

Not Applicable

Related Patterns

Attribute, Random Attribute, Resource

Example Structure



Motivation

Many games assign Attributes to their characters so that various important game aspects of the character can be gauged. The reasons for assigning Attributes to characters is sufficiently covered in the Attributes design pattern, and will not be repeated here. However, the process by which attributes values are set is another important consideration, because attributes can play a key role in any game incorporating them. The Point Spend Attributes design pattern does this by providing a resource (or resources) to a player that he can spend in setting his character's attribute values. So, a player must make character-defining decisions about what is important to him. This process allows a player to quickly and efficiently design a character that he will personally enjoy playing.

Applicability

There are essentially two schools of thinking in terms of character generation:

1) Character generation should focus on creating characters that players are excited about playing,
2) Character generation should focus on creating characters that seemingly existed in the game world prior to being adopted by a player.

Use the Point Spend Attributes pattern when your design goals favor the first option over the second. If you prefer the second alternative, you might want to consider setting attributes by using the Random Attribute pattern instead.

Consequences

From the viewpoint that a character is fundamentally a tool by which a player interacts with the game world, character generation boils down to nothing more than a process by which that tool is constructed. Character generation, then, should be geared toward creating as effective a tool as possible as efficiently as possible. In this light, it is not difficult to see the benefits provided by the Point Spend Attributes pattern. With little fuss, it accomplishes essentially three goals simultaneously:

1) It gives players great flexibility in designing the characters they envision,
2) A Point Spend system does not inadvertently favor one character over another (as can happen when attribute values are randomly generated),
3) It is fast and simple, so it does not unduly extend character generation.

The drawback to a Point Spend system is that it is obvious that the whole process is contrived to allow a player to *design* a character rather than *discover* one. Techniques that randomly generate attribute values and other characteristics are often viewed by players as more of a discovery process, since the player has no real control over what is generated (or his control is at least reduced). The problem with random character generation is that players will often spend inordinate amounts of time rolling dice to "discover" a character they would like to play. The Point Spend Attribute pattern cuts through all that unnecessary red-tape and allows a player to get the character he wants to play right away.

Implementation Concerns

In using the Point Spend Attributes pattern, you should keep it simple. One good way is to create a resource for players to spend solely on setting the values of a group of attributes. The best way to do this is to have one point of the resource equate to one attribute point (see the Currency pattern for reasons why this is so).

Some games allow the attribute resource to be spent on other things as well (such as gifts, skills, or traits). However, doing so can be problematic because this forces the game designer to find some equitable balance between an attribute point and everything else the resource can buy. Otherwise, the choice of what to buy will really be no choice at all since the player will naturally want to gain the most advantage out of the resources you give him.

For example, attributes are often used as adjustments to skill rolls and attributes are generally much fewer in number than skills. In this case, a single attribute point is worth more than a single skill rank because a single attribute is applied to multiple skill rolls. This difference in value increases as the number of skills provided by the game increases. So, a shrewd player will favor attribute points over skill ranks if the same points are used to buy both.

Even worse, if you inadvertently allow a gift that was bought to be re-exchanged for the original resource at a different rate, you have a potentially horrid situation where a player can essentially generate however many resource points he wants. (Don't laugh! At least one version of GURPS has this problem.) If you want to allow players to buy skills, gifts, and other characteristics, there is nothing wrong with creating a separate resource devoted to that end. Isolating subsystems from one another in this way solves the problems mentioned above.

The Point Spend Attribute pattern has one more potential problem. Players that min/max their Characters tend to distribute their points similarly in games that use Point Spend Attributes, at least for similar archetypes. For example, suppose you design a

fantasy game with attributes of both Strength and Magic where Strength is good for hand-to-hand combatants and Magic is good for spell-casters. If you use the Point Spend Attribute pattern, then almost any min/max-er creating a Wizard character will focus his points on the Magic attribute. You will rarely, if ever, see a Wizard with a high Strength and low Magic even though this variant may be interesting from a role-playing perspective. Depending on the importance of these attributes on the mechanics of the game, you may actually end up seeing *exactly* the same distribution of points from character to character.

You may decide this is not a big issue for your game design, though, since it may take a few campaigns and several years for any single group of players to figure out the nuances of your game sufficiently to fall into predictable spending patterns. On the other hand, you may decide that the problem is important enough to mitigate. One potential option is to have some random factor included in the number of points to spend. This would prevent characters of similar archetypes from always having exactly the same distribution of attribute points. But, this solution inherently starts different characters off on different footings, which will likely seem unfair to some players. Another option is to make your attributes conflicted in some way (see the Conflicted Gauge pattern) so that higher attribute values are good in some ways, but are detrimental in others. That way, each player will have to weigh the good and the bad for each point he spends. Different players will make different decisions when confronted with real choices like these. Conflicted Gauges require careful scrutiny by players, though, so making your attributes conflicted will tend to slow down character generation. In short, no solution is perfect.

Samples

Suppose we create a “Top Gun” style game. We plan on making dog fights a big part of the game. To that end, we give the pilot characters attributes such as “Reflex,” “Eyesight,” “G Forces,” “Recklessness,” and “Courage,” all of which range in value from 1 to 10. Conflict resolution uses a number of d10 equal to a skill rank and compares the rolled values against an attribute value. Any numbers that come up greater than or equal to the pertinent attribute value are counted as successes and the side with the greatest number of successes wins the contest. So, low attribute values are better than high attribute values. The attribute values are set by spending a resource of 20 points named “Attribute Bonuses.” All attributes start at a value of 10, but every “Attribute Bonus” spent lowers an attribute value by one to a minimum of two.

Known Uses

Ars Magica gives characters eight primary attributes of “Intelligence,” “Perception,” “Strength,” “Stamina,” “Presence,” “Communication,” “Dexterity,” and “Quickness.” The game has two ways to set attribute values. One of them is a Point Spend Attributes system. Players are given 7 points to spend on attribute values. The costs of a given value depend on a table lookup. Players can earn more points to spend by accepting negative attribute values in some areas.

GURPS characters are designed using a system based purely on "Character Points." Character Points are a resource that players spend to customize their characters. The same resource is used to purchase attribute values as is used to purchase Skills, "Advantages," and other characteristics. Further, more "Character Points" can be obtained by accepting various "Quirks" and "Disadvantages." The cost of any given Attribute value is variable and depends on a table lookup.

Hero System 5th Edition uses eight primary attributes. Their values are specified by having players spend "Character Points" and the costs vary from attribute to attribute. The genre of game, whether heroic or super-heroic, determines the number of points distributed to players to generate their characters.

Inspectres characters have four primary attributes: "Academics," "Athletics," "Technology," and "Contact." "Normal" characters (i.e. humans) are given 9 "dice" to distribute to these four attributes, but all must lie in the range of 1 to 4. (Although the resource name of "dice" implies that dice are rolled to determine the attribute value, the resource is really just a Point Spend system where one resource point equals one attribute point. The Attribute value determines how many dice are added to conflict rolls when the attribute applies, and this is where the resource name originates.) "Weird" characters (i.e. werewolves, vampires, etc.) are given 10 "dice" to spend, with each attribute value falling into the range of 0 to 10. The game allows only one weird agent per game.

Shadowrun characters have eight primary attributes, which are set by spending from a pool of Attribute Points. The size of the pool is determined by the priority at which a player sets his attributes for his character in comparison to skills, magic, race, and resources. So, the more a player emphasizes attributes over other factors, the more attribute points he gets to spend in designing his character. A player may increase this pool size further by accepting "Allergies", which are handicaps of various sorts.

Random Attribute

Intent

Provide a means to generate attributes values within a game environment through the use of die rolls or other random means.

Also Known As

Rolled Attribute, Fortune-Based Gauge, Generated Gauge

Related Patterns

Attribute, Conflicted Gauge, Point Spend Attributes, Resource, Trait

Motivation

The Random Attribute pattern is a specialization of the Attribute pattern where attribute values are generated by some random means, usually dice rolls. The purpose of attributes is sufficiently covered in the Attribute pattern, though, so we won't go into the motive of adding attributes to a game. This discussion focuses on the motivations and concerns of determining attribute values randomly.

This pattern has fallen out of favor in most modern games, although it does tend to crop up from time to time. (That doesn't make the pattern inherently inferior to the alternatives, just out-of-fashion. Its use or avoidance should depend purely on your design goals.) A game designer might use this pattern if he felt that players needed to feel that they are "adopting" a character that "already exists" in the game world rather than having been dreamed up by the player himself. The attribute values are generated randomly to emphasize the fact that the player has no control over his character's make up. Life is random and so is your character.

A game might use random attributes to minimize the number of decisions a player must make and/or to reduce the number of concepts a player must grasp in creating his character before play begins. "You have these seven attributes. Don't worry about what they mean right now, just roll 4d6 and write down the total for each."

Applicability

Use the Random Attribute pattern when you want to provide a fast and easy way to generate attributes and other gauges for characters with little or no player input.

If you use this pattern, you should be aware of the reasons you have decided to do so. The means by which gauge values are set has a profound influence on the playability of any role-playing game. If you are using the pattern merely because you have seen it in other games and are familiar with it, take some time to familiarize yourself with other ways to assign gauge values (such as the Point-Spend Attributes pattern). You may find

the alternatives to be more suited to your game and it would be well worth your time to understand them.

Consequences

The primary benefits of generating attribute values randomly is to speed character generation and reduce the amount of knowledge a player must know about his character before starting play. If your game is designed for "one-shots," or games which end in single sessions where players are unlikely to repeatedly play the game, the time needed to create a character is an important consideration. Most players don't want to learn a lot of complex rules without some reasonable expectation of a future payoff. On the other hand, if the gauges are to have any meaning at all, the players will eventually have to learn their purpose anyway.

Role-playing games that last many sessions are likely to be harmed by randomly rolled attributes rather than aided by them. The Random Attribute pattern does a poor job at gearing a character toward a particular concept. The best the pattern can do is produce random character stats. Since the entire purpose of any role-playing game is for the players to have fun, the adoption of any pattern that does a poor job of enabling the fun to ensue is dubious.

Some games try to gear randomly rolled attributes toward character concepts by first requiring the players to decide on a character "race" or "class" (see the Class pattern). The rules for attribute generation are then focused on producing attributes that are appropriate for that profession. Such systems at least have a better chance of creating playable characters, but completely destroy any arguments about using the pattern to "simulate" a character that "already exists." After all, if a player chooses a character's profession using such a system, then he has provided input into his attributes as well.

Some games try to mitigate the Random Attribute pattern's limitations by allowing players to randomly generate sets of numbers and then have them assign those values to whatever attributes they choose. If your primary motivation for using Random Attributes is to speed character creation, these kinds of modifications *immediately* lose the benefits provided by the pattern. In order to decide how attribute values are to be assigned, a player must understand the meaning of the gauges and must take time to apportion the values to match his goals. No more time needs to be spent on assigning gauge values using the Point-Spend Attribute pattern, and the player gains far more control over his character using that kind of system.

Other games allow players to generate multiple "sets" of attribute values and pick the best set for his player to use. Again, this wastes a huge amount of time and negates the primary argument of using the pattern in the first place.

Implementation Concerns

In using the Random Attribute pattern, you should keep your primary goals in mind because it is very easy to create rules that compete with the pattern's main benefits.

Manipulating the attribute values fights against both the "simulation of reality" and "speed of character generation" goals. So, perhaps the best option is to keep it simple:

1) Decide on what attributes the game is going to use;
2) Select the attributes whose values are going to be randomly generated;
3) Figure out how those attribute values are going to be generated (are they going to be generated differently depending on class or race?);
4) Have players generate the attributes sequentially and take whatever results they get the first time.

After a character is initially generated, you may want to allow modification of character attributes over time to allow for player input as play continues. This can help to alleviate concerns about players being stuck with unplayable characters. (Once again, though, if you decide player modifications are necessary, you might be better served with the Point-Spend Attributes pattern instead.)

Samples

A game might define attributes of Strength, Health, Dexterity, and Intelligence to all characters. Attributes could then be assigned by repeatedly rolling 5d6, summing the dice, and applying the results to each attribute.

Known Uses

Dungeons & Dragons v.3.5 has 6 primary attributes of Strength, Dexterity, Constitution, Intelligence, Wisdom, and Charisma. These are generated by rolling 4d6 six times. In each case, the lowest die is disregard and the remaining 3 are added. The sums are recorded on a piece of scrap paper. Once they are generated, the player may assign the numbers to each attribute however he wants. The numbers are then modified based on the character's race as chosen by the player. If the generated numbers end up being too low (based on defined criteria of what "too low" means), the player is allowed to re-roll his scores. Players are periodically given bonuses to add to their character's attributes as they gain levels (see the Level pattern). Point-spend generation is also standard in many versions, particularly the national tournament level.

RIFTS has 8 primary attributes of Intelligence Quotient, Mental Endurance, Mental Affinity, Physical Strength, Physical Prowess, Physical Endurance, Physical Beauty, and Speed. Each attribute is generated by rolling 3d6 and summing the values. If the result is a 16, 17, or 18, another d6 is rolled and the result is added to the total.

Warhammer Fantasy Role Play has 14 "Characteristics" of Movement, Weapons Skill, Ballistic Skill, Strength, Toughness, Wounds, Initiative, Attacks, Dexterity, Leadership, Intelligence, Cool, Will Power, and Fellowship. Attributes are generated randomly using different formulas based on the character's race, which can be Man, Elf, Dwarf, or Halfling.

Skill

Intent

Provide a means for players to customize their characters with well documented abilities and allow those abilities to improve as play progresses.

Also Known As

Not Applicable

Related Patterns

Attribute, Level, Rank, Resource, Skill Tree, Trait

Motivation

Many games seek to allow each player the opportunity to customize his character's abilities so that the player controls the manners in which he contributes to the overall group. One way to do this is to provide lists of well documented abilities from which players can either choose directly or indirectly through other means (see the Class and Template patterns). These abilities are often called "skills." What distinguishes a "skill" from a "gift" (see the Gift pattern) is that skills generally improve in some fashion over time as play progresses while gifts generally do not.

Having detailed write-ups of the capabilities and limitations of each skill helps reduce the ambiguity of each character's role, because their capabilities are explicitly defined. Further, skill lists help players to build a mental picture of "what's reasonable" within the game environment, because they have material that supposedly facilitates the kinds of actions envisioned by the game designer as allowable.

Applicability

Use the Skill pattern when you want to:

1) Allow players to differentiate the abilities of their characters from others in their group,
2) Allow those abilities to improve as play progresses,
3) Provide detailed descriptions of what each of those abilities accomplishes.

Goals 1 and 2 can be satisfied by the Trait pattern. If detailed write-ups of character abilities are something you want to avoid, you should consider using that pattern instead.

Consequences

The Skills pattern puts a lot of work on a game designer's shoulders. It is one of the major ways in which a "rules-light" game becomes "rules-heavy." On the other hand, it provides a very potent means by which a game designer can direct play toward the

kinds of scenarios he envisions his game to be about. With skills, a game can be made to have a great deal of "crunch." or detail concerning the game's major objectives. However, this comes at the cost of a certain amount of flexibility. Some players find skills based games to be confining regardless of how long a list of possible skill options is provided.

Implementation Concerns

Since one of the primary reasons for using the skills pattern is to differentiate the abilities of different characters, you should take care in designing the skills to ensure that each one plays a unique role within your game. If you fail to do this, players may have trouble in finding meaningful ways to differentiate their characters from one another. For example, a fantasy game in which wizards can easily obtain a spell to open locks would overlap the skills of many "thief"-style characters. In the worst cases, this can lead to a beloved once-potent character becoming completely irrelevant to a storyline and the game will be viewed as "unbalanced."

When implementing the skills pattern, one primary concern is deciding how characters obtain their skills. Do all characters start out with access to all of them with players merely deciding how to customize their characters via some kind of skill ranking scheme? (See the Rank pattern.) Are only some skills available to beginning characters while others must be "earned" as characters progress in the game? (See the Skill Tree pattern.) Are skills granted based on a character's profession or type? (See the Class and Class Tree patterns.) Is some combination of these choices appropriate?

Samples

Karl envisions his new character to ultimately become a James Bond style spy/assassin with excellent sniping abilities. He has 10 skill points to spend on Sam Snyde the Super-Spy. From a list of about 50 interesting choices, he decides to focus on a few basics. He puts 3 skill points into Unarmed Combat, 2 into Handgun, 3 into Sniping, and 2 into Stealth.

A game might specify a "Finding Traps" skill in the following way:

> **Finding Traps**
> Cost to gain rank: 16 experience points.
>
> With Finding Traps, a character can discover the presence of traps on chests, doorways, etc. The character must specifically state where he searches. To succeed, the player must make a normal contest roll using his character's Perception and his rank in this skill. Success indicates the searcher discovers the trap, if it exists. A roll of 1 indicates that the character inadvertently springs the trap.

Known Uses

RIFTS has a list of 125 skills partitioned into the various categories of Communications, Domestic, Electrical, Espionage, Mechanical, Medical, Military, Physical, Pilot, Pilot Related, Rogue, Science, Technical, Weapon Proficiencies, and Wilderness. Each skill is listed with a "Base Skill" rating represented as an apparently arbitrary base percentage (presumably based on the inherent difficulty of performing the skill) plus 5% per level (see the Level pattern). So, proficiency in skills cannot be raised individually, but they do increase in effectiveness as a character gains experience.

TORG has a list of 47 different skills on which "Possibility Points" may be spent to raise their effectiveness individually (see the Rank pattern). Some skills can be used "unskilled" while others cannot. Each skill is associated with a particular attribute (see the Attribute pattern). To use a skill, the "add," or rank, of the skill is added to the corresponding attribute. The result is added to a d20 roll and conflict resolution rules are used to determine success.

Skill Tree

Intent

Provide a means for players to customize their characters with well documented abilities, allow those abilities to improve as play progresses, and expand the number of available options as characters gain proficiency.

Also Known As

Not Applicable

Related Patterns

Class Tree, Level, Rank, Skill

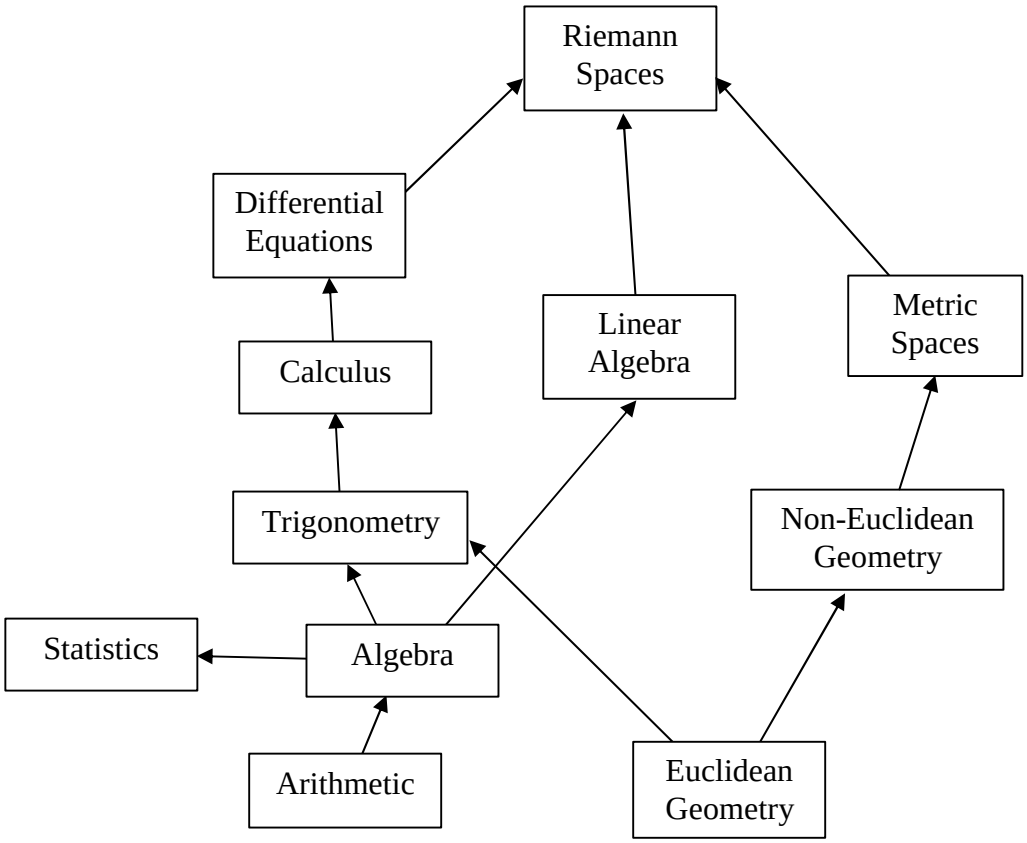
Motivation

The Skill Tree pattern is a specialization of the Skill pattern. It seeks to retain the beneficial characteristics of the Skill pattern (customization of character abilities that improve over time) and also increase the number of in-game options available to a character as play continues. This gradual increase of options is often attained by giving some skills prerequisites that must be met before they can be gained. Commonly, these come in the form of minimum rank requirements in one or more other skills (see the Rank pattern).

The Skill Tree pattern is similar to the Class Tree pattern, in that the growing number of available skill options brings about a qualitative change in a character. Over the course of many sessions, the character doesn't just gain proficiency in the skills with which he started. His skill set actually grows so his range of possible activities broadens.

Example Structure

In a game focused entirely on the one-upmanship of university mathematics students and professors, you might have a skill tree structure similar to the following:



Applicability

Use the Skill Tree pattern when you want to

1) Allow players to differentiate their character's abilities from others in their group,
2) Allow those abilities to improve as play progresses,
3) Increase the range of skills available to a character as play continues,
4) Provide detailed descriptions of what each of those abilities does.

Goals 1 and 2 can be satisfied by the Trait pattern. If detailed write-ups of character abilities are something you want to avoid and an increasing range of character options is unimportant to your game, you should consider using that pattern instead.

Note that the Skill Tree pattern provides no niche protection for character abilities. If you need this feature, you might want to consider the Class Tree pattern. Note that the

Class Tree and Skill Tree patterns are not mutually exclusive, but if both are used, the niche protection characteristics of the Class Tree pattern may be reduced unless care is taken to exclude any niche protected abilities from becoming generally available.

Consequences

The Skills Tree pattern requires a lot of effort from a game designer, because detailed descriptions of all skills must be created. It can easily require even more writing than the basic Skill pattern, because the Skill Tree pattern must provide a sufficiently broad mix of abilities for beginning characters and also give a rich menu of interesting options as characters progress.

The Skill Tree pattern does not restrict characters in the skills they may attain provided they meet the stated requirements of the skills they seek. It is therefore one of the more flexible ability patterns commonly in use. However, its very flexibility means that a seasoned character, who has gained good use of a wide range of abilities, can tend to overshadow less experienced characters. For example, a powerful "Batman" type superhero that has accumulated a smorgasbord of "bat belt" abilities may render a less experienced "Cat Woman" style character impotent by comparison, especially if her abilities are merely a subset the more powerful character's skill set.

One downside of the Skill Tree pattern is that the array of skills available to a character at any given time can be difficult to determine. So, it can become difficult for players to plan out their character's careers ahead of time. This is especially true for beginning players.

Implementation Concerns

The sheer number of options available to players in a game using the Skill Tree pattern may confuse new players to the game. It is therefore advisable to clearly indicate what skills are "beginning" and which ones demand prerequisites. This can be done either by splitting out the "beginning" skills entirely from the "advanced" skills, by providing a conspicuous table of beginning skills to which a new player can refer, or by providing sample "starting packages" typical of beginning characters (see the Template pattern).

If you are not careful in how you implement the "paths" that characters can take from one skill to another, you may limit the expandability of your game unnecessarily. For information on how you can avoid these potential pitfalls, see the Loose Coupling pattern.

Samples

A fantasy game might define its mage spell system using the Skill Tree pattern. In such a game, an "Open Sesame!" spell might be defined in the following way:

Open Sesame!

Lore Requirements: The mage must attain 9th rank in the arcane areas of both Dust and Wind
Experience Cost to Gain Each Rank: 8 points

After this spell is cast over a single doorway, the door holds firmly against anyone who does not speak the password. The initial casting takes 18 minutes, during which the mage must touch the door. However, the charm is permanent thereafter and the door can be quickly opened or closed by a password. The password, spoken by the mage when the spell is originally cast, can be any combination of sounds desired. The door opens when the words "Open" followed by the password are spoken. The door shuts again when the words "Close" followed by the password are spoken. To force the door open, a normal contest roll must be made between the aggressor's Strength and the spell rank. The minimum Margin of Success needed to force the door open equals triple the spell rank. Opening or closing the door by force permanently negates the spell.

Known Uses

Dungeons & Dragons v.3.5 uses the Skill Tree pattern for its magic system, although at first glance you may have difficulty recognizing it as such. The selection of spells available to a spell casting character increases as he gains "levels" (see the Level pattern), and the potency of the spells tend to increase in power along with level increases, so it does indeed fit the pattern. For each class (see the Class pattern), the game partitions its spells into 1st through 9th "level" spell categories. The spell categories that are available to a character *depend* on the character's "level," but the spell levels available to a character and his character's level are not equal, although they are correlated by a table.

RIFTS has a list of 125 skills partitioned into the various categories of Communications, Domestic, Electrical, Espionage, Mechanical, Medical, Military, Physical, Pilot, Pilot Related, Rogue, Science, Technical, Weapon Proficiencies, and Wilderness. Each skill is listed with a "Base Skill" rating represented as an apparently arbitrary base percentage plus 5% per level (see the Level pattern). So, proficiency in skills cannot be raised individually, but they do increase in effectiveness as a character gains experience. Some skills have a "Requires" section that describes the skills that a character must already possess in order to gain the skill. For example, the skill of "Mechanical Engineer" demands the character to have "Basic or advanced mathematics, at least basic electronics, and literacy."

GURPS (Basic Set) has a list of approximately 174 skills (depending on how they are counted). "Bonus points" can be spent on raising skill "levels" (see the Rank pattern). Some skills have "Prerequisites," which are usually other skills. To gain a skill with a prerequisite, a character must already possess the required listed skill at a "skill level" of 12 or more.

Template

Intent

Provide a means to quickly assign a group of abilities to a character.

Also Known As

Package Deal

Related Patterns

Class, Gift, Skill, Trait

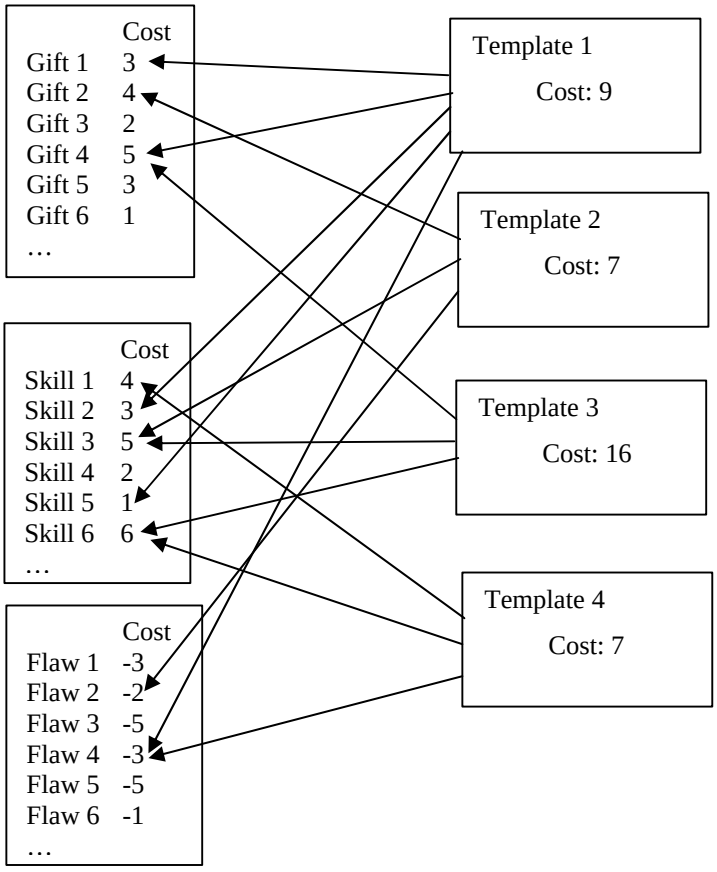
Motivation

A template is a collection of flaws, gifts, traits, and/or skills that are given to a character when his player adopts the template. Often templates are used in games where skills and traits can be "bought" by spending some resource provided for that purpose. In such games, different templates often consist of nothing more than a pre-packaged group of abilities whose total costs have already been determined. "Buying" a template is equivalent to buying all of its individual abilities. This application does not prevent the character from buying other abilities in the future if he has the resources to do so. Once a template is applied to a character its utility may be finished. In such cases, the player need not even record the fact that it was used.

Though rare, it is possible to use templates in trait-based games as well. By definition, traits are not pre-defined (if that makes sense). But, templates incorporating traits can still be designed if they merely act as examples of how traits might be put together.

Templates are often used in place of classes (see the Class pattern). They gain the Class pattern's advantage of rapid character creation while avoiding the inflexibility produced by the Class pattern's niche protection. Whereas classes restrict a character on the kinds of abilities he can possess, templates do not.

Example Structure



Applicability

Use the Template pattern when you

1) Want to quickly provide pre-designed groups of abilities to characters.
2) Do not want to restrict characters in the kinds of abilities they can gain.

While character restrictions sound like a bad idea at first, they can actually benefit games having characters with similar abilities in common. This can happen, for example, in a game where all of the characters are thieves. Players could choose from a number of pre-defined thief types such as Safe Cracker, Con Artist, Cat Burglar, Fence, Rogue, Thug, and many more. Each of these thief types could have a unique and important role to play in an underworld style game. However, without some restrictions, character distinctions in a game of this kind could quickly blur. If you want to ensure that characters maintain meaningful niches in your game, you might want to consider the Class pattern instead.

Consequences

The Template pattern pretty much accomplishes what it sets out to do. It doesn't complicate a game, because all it really does it package groups of abilities for easy consumption. It also doesn't interfere with the rest of the game design, because templates are ordinarily "disposable" game mechanics. Players usually use them once and forget about them (there are exceptions, most notably in trait-based games).

Templates are not a replacement for classes, merely an alternative. Since templates don't really impose restrictions on characters, they also don't provide a game designer with any means to keep characters from "stepping on each others' toes." No character can truly claim any role as his own "territory", because any other character could gain his abilities any time he gained enough resources to "buy" them. Consequently, players can have a more difficult time distinguishing their characters from others in their group.

Implementation Concerns

Since templates are essentially pre-packaged groups of abilities, you need to decide how characters gain access to those packages. One very common technique is to assign each character a resource that he can spend on gaining skills and templates (see the Resource pattern).

If you want to encourage the use of templates over the purchase of individual skills for some reason, you might want to consider package deals. "Buy 10 skills, get one free!" In other words, make the total cost of the template slightly less than the sum of the abilities it grants.

Samples

The following might be an example of a template in a game containing middle-age thieves in which the Skill Rank pattern is used:

Bandit

Cost: 20 Development Points
Bandits often group together to form ambushes on caravans and wealthy nobility. They constantly try to invent new ways to trap and overcome opponents normally considered too powerful to defeat. Of course, they frequently set up their surprises in ravines and mountain passes, but imaginative ploys always inspire these thieves. They realize that only a limited number of ambushes are safe at a given spot before some *real* force shows up.

Skills: Stealth (rank 2), Setting Traps (rank 1), Climbing Walls (rank 1), Horsemanship (rank 2), Tracking (rank 1)
Weapon Proficiencies: Sword (rank 2), Firing crossbows (rank 1)

Known Uses

Hero System 5th Edition has "Package Deals" that are essentially groups of skills, gifts ("Perks"), and disadvantages that cost "Character Points." Game Masters are encouraged to come up with their own package deals, which can be professional ("policeman") or racial ("elf"). In general, package deals cost a character fewer Character Points than if he had purchased all of the characteristics individually.

HeroQuest has "Keywords" that act as a sort of trait templates. Keywords can be used to describe a profession, homeland, species, or magical proficiency. They are essentially groups of traits that all share the same "rating" (see the Trait and Rank patterns). Keywords generally have a rating of 17, but there are exceptions (most notably for species). These ratings cannot normally be improved as play progresses, although there are optional rules that allow this. Through play (and player agreement), keywords can be altered or even eliminated over time.

TORG has rules for creating templates, and provides a number of pre-defined templates in supplements. Essentially, every new character has 66 "attribute points" to spend on attributes (see the Point-Spend Attributes pattern) and 16 "skill points" to spend on "skill adds" (see the Rank pattern). Magical characters get an additional 12 skill points to put into "Arcane Knowledges." The templates distribute these points in a rational fashion according to technology level, race, culture, etc. Since these templates spend all of a beginning character's resources, the player simply decides which one he likes and sets his stats accordingly.

Trait

Intent

Provide a flexible means to specify a character's abilities within a game environment without requiring the game to provide a pre-defined list of abilities.

Also Known As

Not Applicable

Related Patterns

Attribute, Rank, Resource, Template

Motivation

Traits provide a flexible way to describe a character's ability to influence the game world in various ways. Traits are generally made up on a character-by-character basis such as "Loves Chocolate" and "Expert in Computer Programming." Numbers are often assigned to traits to specify how much better or worse a character is likely to perform on a given action as compared to the "norm" (see the Rank pattern). In some games, traits are specified when a character is first generated. In others, traits are not pre-set, but are added to characters as the game progresses. What is important is that the traits provide a means to describe how well a character is likely to perform on any given action within the game environment when conflicts arise.

The flexibility in assigning what traits a character possesses allows a player to state what he feels is important to his character. It also provides a way for him to gently guide the game toward the kinds of scenarios in which he is interested, because his character's traits will only be applied if the character behaves in a way that bring those traits into play. This encourages the player to seek out scenarios in which his character's important traits will likely apply, and so the game will naturally tend to gravitate toward circumstances in which they *do* apply.

Applicability

Use the Trait pattern when you need to provide definite guidelines on how character abilities will influence the resolution of conflicts, but feel flexibility in character customization outweighs the need to pre-specify fixed lists of skills, and/or classes. If having a pre-determined set of character classes would enable you to meet your game goals better, you might want to consider using another pattern instead.

The Trait pattern does allow new traits to be added as characters develop, but it provides no well-defined means of how traits relate to one another. If you want your game to reserve some abilities as a sort of incentive for players as play progresses, you might be better served by the Class Tree or Skill Tree patterns.

As a general rule of thumb, the more flexible a system becomes, the harder it is to implement in a computer. So, if you are designing a role-playing game with the intention of implementing it in software as an interactive video game, it is highly recommended that you use a different pattern for character abilities. The Trait pattern borders on impossible to write in software with current programming techniques.

On the other hand, games using the Trait pattern tend to be less verbose in their overall description since they forego the need to create pre-determined lists of abilities. So, if you seek to write a "rules-light" game, the Trait pattern is ideal.

Consequences

The Trait pattern provides a highly flexible means by which players can customize characters and still allow character interaction with the game world to be gauged. Players that find pre-set lists of options to be limiting will likely find games using the Trait pattern to be quite liberating. This comes at the cost of introducing a degree of vagueness of when a particular trait does or does not apply in any given conflict. In tactical games where players compete with one another, this ambiguity could cause problems. Without clear rules governing how such disputes are to be resolved, this ambiguity can tend to slow games down with repeated discussions of what traits apply on a conflict-by-conflict basis. In non-tactical games, though, arguments concerning the applicability of traits rarely arise.

Implementation Concerns

The Trait pattern can introduce difficulties in deciding what exactly each custom trait really means and how they interact in conflicts. Such a system must pay particular attention to the descriptions they provide of what constitutes a valid trait and how they affect one another in game play. In cases of dispute, the game should provide clear rules on how to determine when traits apply. This can be as simple as assigning one player to be an impartial "judge," such as a Game Master.

Games using the Trait pattern often apply numerical values to the traits to give different characters with similar traits an additional means of customization. If this kind feature would enhance your game, you might want to consider applying the Rank pattern to your traits.

One problem with traits is that broadly defined traits can be applied to a wider range of situations than narrowly defined traits. What this means is that players are encouraged to define traits in as broad a terms as they can negotiate with their Game Master or other players (i.e., "Jack-of-all-Trades" vs. "Auto Mechanic"). But, narrowly defined traits push players to be more creative in their use and tend to give characters more personality. In short, narrow traits often enhance the role-playing experience to a greater degree than broad traits. So, if you use Traits in your game, you might want to consider instituting some method to encourage the use of narrow traits or to otherwise "balance out" the use of any trait.

Some ideas on how do to this are

1) Require a player to spend some resource whenever he uses a trait. For example, make a player spend one "Plot Point" every time he employs one. This makes broadly defined traits cost more than narrowly defined ones.
2) If you are using ranked traits, make the cost of gaining ranks in the trait tied to how broad or narrow it is. So, for example, a "Jack-of-all-Trades" trait would cost 5 Character Points per rank while "Auto-Mechanic" costs only 1. Of course, this option requires the players to negotiate in advance how broad or narrow the trait is and set the cost accordingly. So, it is still somewhat subjective.
3) Make traits conflicted (see the Conflicted Gauge pattern). One way to do this is to allow a trait to be used as a "bonus" in a game only after it has been used as a "penalty." For example, a "Death Wish" trait could be used to help a cat burglar leap from one rooftop to another while running from pursuing cops. But, it could only be brought into play if the character was previously in a scene where it hindered him in some way, such as resisting the suggestion to play Russian Roulette with an old enemy.

Samples

In a "Wind in the Willows" style game, one player decides to create an anthropomorphic frog character named "Toady" that loves to sit by the fireplace, smoke his pipe, and read books. To do this, he assigns a general trait of "Frog" to his character, expecting that to provide a number of "froggish" abilities, such as swimming and bounding. He also assigns the trait "Well Read" and "Loves Tobacco." Later, the nearby swamp rat villain "Ratatask" invades his storehouse and absconds with his goods, including his tobacco supply, leaving ample muddy paw prints on his otherwise spotless floor. Toady's player asks that his character's "Well Read" trait be applied to finding clues as to where a swamp rat might take his stolen goods, to which his GM agrees.

Known Uses

HeroQuest has "abilities" that are not pre-defined by the game, and so match the Trait pattern. Very broad abilities are known as "keywords" that act as templates (see the Template pattern), and represent all abilities common for an occupation, culture, religion, and the like. Each ability has a proficiency "rating" (see the Rank pattern).

The Pool uses "traits" to describe character abilities. In this game, a trait is associated with a numerical value that allows a player to add dice to his character's "Dice Pool" in resolving conflicts involving the trait (see the Rank pattern).

Universalis allows players to apply traits to characters through the expenditure of "coins," such as "Loves Children" or "Blacksmith." Even a character's name is a trait in Universalis that must be purchased through coin expenditures. The more coins that are spent on a particular trait, the greater its "importance" rating (again, see the Rank pattern).

Trauma Gauge

Intent

Provide a gauge for harmful effects on characters that hinders the use of related attributes and abilities.

Also Known As

Health Level, Weariness

Related Patterns

Endgame, Hit Points, Wound Trait

Motivation

Many role-playing games deal with exhaustion and mental or bodily injury. Games that want to incorporate the incremental effects of various forms of injury, whether physical or emotional, need some means to adequately simulate these effects without bogging down the game mechanics with too much minutiae. Any role-playing game that assigns a gauge to a character that somehow measures the character's current state of health (whether physical, spiritual, mental, or otherwise) and uses that gauge to lower the effectiveness of related actions follows the Trauma Gauge pattern.

In its simplest form a Trauma Gauge has a value indicating the current degree of wounding that a character has suffered. This gauge hinders the effectiveness of all related abilities the character attempts. Often, a Trauma Gauge's value is directly subtracted from a character's effectiveness although this is not a requirement of the pattern. If a gauge measures the level of wounding a character has sustained and it has some increasingly detrimental effect on a character's actions as its value increases, it follows the Trauma Gauge pattern. So, the greater the gauge value, the more the character suffers from his injuries. Note that the pattern does not require that there be any pre-defined level of injury at which the character "dies." Depending on the game requirements, a character could continue suffering greater and greater quantities of trauma forever. In this way, the Trauma Gauge pattern greatly differs from the Hit Points pattern. Of course, there is nothing preventing a game designer from "mixing in" the Hit Points pattern by setting a limit to the amount of injury a character can suffer before expiring.

Note that any gauge that measures a character's state of "Health" is, for all practical purposes, equivalent to any attribute measuring a character's state of "Ill Health." We have chosen to call this pattern the Trauma Gauge pattern rather than the Health Gauge pattern to highlight its differences from the Hit Points pattern. If you use an attribute gauging the level of "Health" rather than "Injury," you could easily state that a character dies if his Health drops to zero. To have a Health attribute modify character actions, then, you would first have to determine the level of damage sustained before applying any adjustments (assuming a fully healthy individual applies a zero adjustment to his abilities).

Applicability

The Trauma Gauge pattern is appropriate in games where you

1) Need to incorporate rules dealing with injury and possibly death.
2) Want to simulate the incremental effects of injuries on character actions.
3) Are willing to accept the added calculation demanded by the pattern.

If you find the arithmetical overhead too high a price to pay, you might want to consider the Hit Points pattern instead. On the other hand, if you are willing to live with even more bookkeeping to provide finer control over wounding effects, you should consider using the Wound Trait pattern. Finally, if you want to guarantee the survival of characters until their roles are fully played out in a storyline, you should consider the Endgame pattern.

Please note that the Trauma Gauge, Wound Trait, and Endgame patterns are not mutually exclusive. So, you might want to contemplate using some combination of these patterns in your game rather than rely on any one pattern in isolation.

Consequences

The Trauma Gauge pattern provides a gauge of a character's current level of damage and modifies the success of related actions based on this state of health. While this sounds like an unquestionably Good Thing at first, the pattern does suffer from a potential problem known as a "Death Spiral." If a character is wounded in a game incorporating the Trauma Gauge pattern, his abilities are henceforth reduced in potency. This makes it easier for the character to sustain further injury, since any abilities he might use to avoid future wounds are hampered. Sustaining more damage means the character is even more susceptible to damage, etc. Unchecked, the pattern nudges a character further and further along a spiral toward his death (or similarly catastrophic result).

Generally, Hit Points are only modified and considered when a character sustains some kind of damage. A Trauma Gauge, on the other hand, must be considered on every character action to

1) See if it applies to the action, and,
2) Determine how much it affects the action.

Assuming characters perform actions more often than they sustain damage, the Trauma Gauge pattern requires more overhead. Obviously, systems can be devised that reverse this tendency. But, in most cases, Trauma Gauges require more overhead than Hit Points.

Implementation Concerns

If you use the Trauma Gauge pattern, you will want to make sure that the gauge is applied uniformly across all related abilities so that it does not unfairly favor or hinder any particular character actions. One way to do this is to ensure that all character attributes lie on the same "scale" and all abilities use the same mechanics to resolve conflicts. In other words, don't make your character's Strength attribute range from 1 to 4 while his Agility attribute has a range of 1 to 100. Similarly, don't use a d12 for Stealth checks and percentile dice for combat attacks.

To reduce the "Death Spiral" effect, some games apply an adjustment *based* on the Trauma Gauge to character actions rather than use the gauge value directly. The adjustments give a nod toward the debilitating effects of wounds without allowing them to overpower a character too quickly. So, a Trauma Gauge value of 10 might translate to a penalty of -2 and a value of 20 might translate to an actual penalty of -4.

Another option you might want to consider is to allow a player to "spend" his injuries by accepting temporary (or permanent) flaws in exchange for a reduction in his current Trauma Gauge value.

Samples

Consider an "Office Cubicle" game that has an "Emotional Stress" attribute specifying the current level of emotional injury that a character suffers. Conflict resolution is performed using 1d10, adding pertinent abilities, subtracting "Emotional Stress" and comparing the result to a threshold. The lists of traits and attributes include subjects such as "Office Gossip," "Backstabbing," and "Political Correctness." The more meaningless paperwork piled on a character and the tighter his deadlines, the more emotional damage he sustains. "Say, Wilson, do you have those TPS reports done, yet?" Take 1 Stress. The character's work ethic and productivity dwindle as greater quantities of stress are piled on. Fortunately, at any time, a character can decide to "spend" his stress points by indulging in some stress relieving activity:

Activity	Stress Relief
Jogging	-1 Stress
Make fun of the Boss	-1 Stress
Vacation	-2 Stress
Report Anonymous Tip to IRS about Company Accounting Practices	-5 Stress
Quit Job	-5 Stress
Writing a program to Embezzle Funds	-8 Stress
Burning down the Office Building	-10 Stress
Going "Postal"	-10 Stress

Known Uses

In **My Life with Master,** minions have a "Weariness" attribute that increases as they suffer injury. Weariness is directly subtracted from all actions except when approaching "Connections." Connections are NPC's that enable minions to gain "Love" which is an attribute that allows a minion to resist his Master's commands.

The Riddle of Steel has a Pain attribute that is subtracted from a character's dice pools on every combat round. Pain accumulates as wounds are inflicted. The more Pain that a character suffers, the less effective he is in combat and magic.

Sorcerer has a Damage attribute (although it does not appear on the character sheet). This attribute accumulates all damage inflicted upon a character and is directly subtracted from all actions attempted by the character. Players have the option of having their characters make "Will rolls" without penalty to temporarily forego some or all of the effects of Damage. (This nicely sidesteps the death spiral effect.) The game specifies the physical effects of different levels of Damage in comparison to a character's "Stamina," such as "bruises," "need stitches," and "guts hanging out." But, interestingly enough, it doesn't explicitly specify that a character can die from damage alone. Death can arise when a character's "Humanity" attribute drops to 0, but alternatives to death exist as well depending on what Humanity actually means to a given character.

The World of Darkness (Werewolf: The Apocalypse) gives each character a "Health Level" attribute. After a successful attack is made, a separate damage roll determines the amount of damage delivered. The game uses a dice pool for conflict resolution, so each success on the damage roll indicates a point of damage. The Health Level attribute has 7 possible values (8 if you count "Uninjured"). These are: Bruised (-0), Hurt (-1), Injured (-1), Wounded (-2), Mauled (-2), Crippled (-5), and Incapacitated. Werewolves regenerate damage at the rapid rate of one point per combat round, except for wounds delivered by silver, fire, or other supernatural creatures.

Wound Trait

Intent
Maintain injuries, physical or otherwise, as individual character traits.

Also Known As
Not applicable

Related Patterns
Endgame, Hit Points, Rank, Trait, Trauma Gauge

Motivation
Games that individually track character injuries follow the Wound Trait pattern. In its simplest form, the pattern does not provide any mechanical means of modifying a character's abilities when harmed. It merely describes the specific injuries a character has sustained. If wounds need to impose some mechanical restrictions on characters to satisfy design goals, they may certainly be augmented with other patterns, such as the Rank, Hit Points, and Trauma Gauge patterns.

One reason a game designer might utilize wound traits is to enhance story potential. If a player knows that a character has a gaping flesh wound on his left thigh, he can take that into account when he narrates scenes involving the character. Games having combat as their primary focus are often designed to simulate the "gritty realism" of battle, and may use wound traits as one means of doing so.

Games about combat realism are likely to mix this pattern with the other injury tracking techniques described above. They may also handle injuries far differently than games centered on drama and story. In real life, wounds are a Bad Thing, so a game designer seeking to accurately reflect real world conflicts will likely implement wounds as having universally detrimental effects to those injured. The degrees to which various wounds hinder characters may vary drastically depending on the location and severity of the wound, but are unlikely to ever aid him.

On the other hand, a game targeting the telling of interesting stories might take a different approach. First of all, wounds might be introduced only if they drive the storyline forward in some fashion. For example, in a game based on the horror genre, a storyline might progress more quickly and with greater believability if the protagonist, a female Olympic track star, conveniently suffers a crippling leg wound so that the serial killer can catch up with her and initiate the game's climax. In such a game, a player might willingly accept (or even suggest) this kind of a wound to increase dramatic tension and push the plot forward. In so doing, the game might reward the player by actually making it *harder* for the serial killer to slay the fleeing character. One good way to reward players that accept disabilities to enhance storytelling is to use an

abstract meta-game mechanic. This maintains the "believability" of a story (in that it doesn't give wounded characters absurd physical enhancements) and at the same time allows players more flexibility in their story options. Our horror game example might provide players with "Plot Points" that they could spend in augmenting contest rolls (see the Resource pattern). Our heroine would be physically hindered by her wound, but the player would be rewarded with enough "Plot Points" to more than make up for the injury's debilitating effects.

Applicability

The Wound Trait pattern is appropriate in games where you

1) Want to track individual wounds to increase combat detail or enhance story potential.
2) Are willing to accept the additional bookkeeping that recording individual wounds demands.

In games where wounding effects occur infrequently, the Wound Trait pattern is ideal. True, the bookkeeping overhead is relatively high on a wound-by-wound basis. But, the cost of tracking a small number of injuries during a game session is generally well within the tolerance of most players' sensibilities given the level of story potential the pattern provides.

If you find the bookkeeping too high a price to pay, you might want to consider the Hit Points or Trauma Gauge patterns instead. If you want to guarantee the survival of characters until their roles are fully played out in a storyline, you should also consider the Endgame pattern.

Consequences

Of all the health tracking patterns, the Wound Trait pattern provides the most detail. It gives players reasonably clear views of their characters' states of health, so scenes in which injuries play a factor become easier to describe. However, overuse of the pattern can bog down game flow with details that add little or no enjoyment to players. So, you should use the pattern only to the degree that it supports the "mood" your game tries to generate. Anything more will detract from your ultimate vision.

Implementation Concerns

On its own, the wound trait pattern is conceptually very simple. If a character is harmed, a note is made of the wound's characteristics. However, if you need wounds to provide some mechanical effect in contests, you need to find some other means to accomplish this goal. Wound traits are commonly mixed with other patterns. For example, wounds can be assigned a Rank to describe their severity. They can be associated with a damage rating, which is subtracted from a character's Hit Points or added to a Trauma Gauge.

Some games provide tables that specify the effects of specific wounds. Wound tables are usually based on the severity of a blow and can take into account factors such as hit

location and armor types. They can be as large as necessary to give the required level of detail. Be warned, though, that table lookups like these can take significant amounts of time to use.

Samples

A game in which characters all play World War II infantry might rank individual wounds according to their severity. The sum of the individual ranks could then be used as a modifier to physical actions. After a difficult battle, a soldier might have the following traits listed on his character sheet:

Sergeant Peppered

Wound	Rank
Grazing bullet wound on left shoulder	1
Shrapnel wound on right thigh	2
Bayonet wound on lower right abdomen	5
Total	8

Known Uses

The Riddle of Steel has a variety of means to simulate injury. First, it has a Pain attribute that is subtracted from a character's dice pools on every combat round. Damage is also delivered in the form of "Shock," which reduces a character's short term combat effectiveness (a single round), but does not linger. Blood Loss also drains Health, which can kill a character if Health drops to zero (see the Hit Points and Attribute patterns). However, perhaps the most important form of damage in the game comes from "Wounds." A wound is an injury delivered to a specific location. The severity of a wound is determined by the Margin of Success of the attacker in landing the blow. Its effects are determined by both its severity and its hit location according to various tables. Each wound is unique with effects ranging from "Charlie Horse" to "Death. Destruction of cerebellum. Really Messy."

Rolemaster Fantasy Role Play uses "Hits" as its primary means of tracking physical punishment (see the Hit Points pattern). But, it also uses wound traits to add flavor and detail to combat. Attacks are made by rolling d100, adding the aggressor's "Offensive Bonus" (OB), subtracting the defender's "Defensive Bonus" (DB), and looking up the result on a table based on the weapon type used to deliver the blow. The tables take into account the defender's armor type. This lookup provides a damage rating, which is subtracted from Hit Points, and a "Critical" rating ranging from A to E. The d100 is rolled again and the result is looked up on a "Critical Strike Table" corresponding to the weapon type used (Puncture, Slash, Heat, etc.). The second table lookup is based on the criticality rating of the previous table lookup. The effects take into account whether the defender must parry, cannot parry, is stunned, bleeds, or has other penalties and for how long. The sheer volume of tables gives a wide variety of possible results ranging from "Not very impressive" to "Blow turns hip to dust. Foe falls down. Attempts to stand. Falls again and dies in 6 rounds."

Universalis treats injuries like all other traits. Wounds are purchased through the expenditure of "coins," which is the sole source of narrative power in the game (see the Currency pattern). Wounds are simply traits that have the characteristic of reducing a character's effectiveness in some way. Multiple coins can be spent on any given wound to augment its severity. Player must take the effects of wounds into account when having characters perform actions. If they fail to do so to the other players' satisfaction, the "importance" of the wound counts against the character in conflict resolution. A character can only be killed or otherwise permanently removed from a storyline by spending a number of coins equal to the characters total "importance." His importance equals the sum of all importance values of all traits. So, a character that has been inflicted with a wound is actually *harder* to kill than one that hasn't been wounded, even though the injury hinders his activity.

Fundamental Gauge Patterns

Conflicted Gauge

Intent

Provide a gauge that aids a character sometimes and hinders him at other times. This forces players to make difficult character defining decisions to balance various goals.

Also Known As

Not applicable

Related Patterns

Attribute, Gauge, Skill, Trait

Motivation

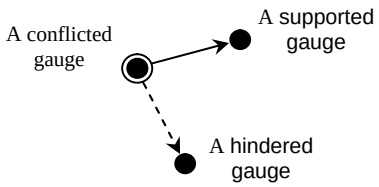
If a game gives characters a gauge where high values are beneficial at certain times and low values are beneficial at others, the game follows the Conflicted Gauge pattern. Ordinarily, the values associated with the gauges are numerical, but this is not a requirement of the pattern. A Conflicted Gauge may be an attribute, skill, trait, or any other gauged characteristic. An attribute that is a Conflicted Gauge is said to be a Conflicted Attribute. Similarly, a trait that is conflicted is said to be a Conflicted Trait, etc.

A conflicted gauge could be set up to reflect a person's power or capability within one domain and, at the same time, represent the character's weakness in another. A game could incorporate a "Rage" attribute that increases a character's physical strength and speed values, but at the same time reduces the character's ability to reason.

Conflicted gauges can also encourage players to voluntarily portray their characters with self-destructive behaviors. A game designer might use a conflicted gauge as a form of "temptation." The game mechanics might encourage a player to take a short and easy route to power or success by allowing him to quickly raise a gauge to high values but, in so doing, force him to accept some detrimental longer-term effect. In such a case, the player has to make a real decision as to whether he should raise the value, lower it, or leave it the same. Short term needs must be balanced against long-term goals. So, a certain amount of angst is naturally associated with the gauge. Setting or changing its value forces a player to make a statement of how much he values the gauge’s benefits as weighed against its eventual costs.

Example Structure

For a gauge to be conflicted, it must benefit a character in one way, and be detrimental to him in another.



Applicability

Use the Conflicted Gauge pattern when your design goals include one or more of the following:

1) A need for characters to exhibit self-destructive behavior at times.
2) A need for particular character strengths to simultaneously exhibit themselves as character weaknesses,
3) A desire to force players into choosing between short term needs and long term goals.

Consequences

The primary benefit of the Conflicted Gauge pattern is that it provides a mechanism whereby a character aspect is both good and bad. It can be used to encourage character actions that are ultimately self-destructive in nature but are, nevertheless, rational role-playing options from the player's perspective. Thus, a player who always seeks to do the "best thing" for his character may end up with the character performing actions with detrimental consequences. The result is that role-playing is enhanced with rational ways of simulating real world irrational human decisions.

Implementation Concerns

Games incorporating Conflicted Gauges essentially force players to frequently make judgments concerning the direction they want to take them (up or down). This means that any conflicted gauge is likely to draw a significant amount of player attention. Those that are incorporated should be tightly focused onto the core premises of the game. Conflicted Gauges make a very strong statement about the game's overall theme and purpose, so don't muddle your game design with extraneous ones.

The Conflicted Gauge pattern states nothing about what the gauge simulates, other than that it must entail two mutually exclusive characteristics. Just because they are opposed, though, does not mean that the beneficial aspects of the gauge must be of the same nature as its detrimental effects. Players will be willing to accept handicaps concerning the gauge that range far afield from its benefits as long as its hindrances seem rationally connected in some way to its augmentations.

Samples

A game about the self-destructive lives of drug addicts could give characters attributes of "Addiction," "Stress," and "Cool." In this scenario, Cool could be the attribute used in inter-personal conflict resolutions, whether social or physical. High values of Cool are always good. Stress would be an attribute that is subtracted from all Cool rolls, and increases whenever conflicts fail (see the Trauma Gauge pattern). So, high Stress values are always bad. Addiction, on the other hand, could be set up as a Conflicted Attribute. If a character is stressed, he can "binge" on his preferred mind altering drug to quickly lower his Stress by an amount equal to his Addiction. Every time he does so, he must make a roll to determine if his Addiction increases by one. However, the

higher his Addiction value, the more expensive his habit becomes as he needs more and more of the drug to induce the Stress relieving effects. If he cannot indulge in a minimal amount of the drug after a number of days equal to one week minus Addiction, he adds one point to Stress every day he goes without.

Known Uses

Call of Cthulhu allows characters to gain the "Cthulhu Mythos" skill. Gaining ranks in this skill gives a character better insight into terrifying secrets of the universe. Such knowledge is highly valuable in surviving encounters with the unspeakable horrors populating the game's setting. However, as a character's rank increases in this skill, his sanity decreases. So, as characters learn more and more about the hopeless reality of the world, they become more and more mentally unstable. Thus, the Cthulhu Mythos skill is conflicted.

My Life with Master has characters who are minions of Evil Masters. All minions have attributes of "Self-Loathing," "Weariness," and "Love." The Self-Loathing attribute is conflicted because it aids minions in performing horrific, monstrous deeds for their masters but, at the same time, hinders them in any attempts at disobeying their Master's commands.

Sorcerer is a game in which characters are humans that summon and bind powerful demons to their will. Each character has a conflicted attribute known as "Humanity," which also follows the Idiom pattern (and possibly also the Resource pattern). Humanity is used in conflict resolution rolls for both summoning and banishing demons. When used to banish demons, Humanity augments the roll. However, the attribute detracts from a character's chances of success when summoning demons.

Currency

Intent

Provide a flow of mechanical influence between gauges in a game system.

Also Known As

Not applicable

Related Patterns

Gauge, Generalized Contest, Negotiated Contest

Motivation

Gauges derive their meaning from the relationships they have with each other. Currency quantifies that meaning. Any two gauges that transfer mechanical influence from one to the other demonstrate the Currency Design Pattern. The transfer usually occurs by adding or subtracting one gauge value to or from another. That does not mean that the originating gauge is diminished by the action, though. Depending on the purpose of the transfer, one gauge may simply add its own value to that of another gauge. (If the originating gauge *is* affected by the transfer, it most likely follows the Resource Design Pattern.)

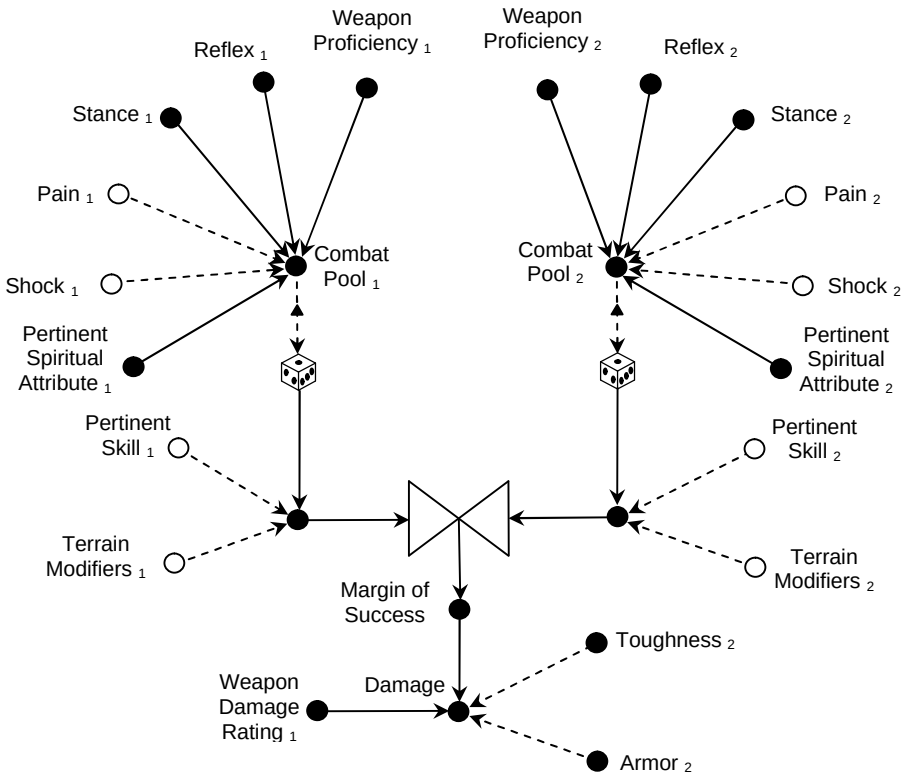
Along with gauges, currency is one of the primary building blocks of role-playing game design. As a game designer, you would be wise to gain a firm grasp of its potential and pitfalls. To create a successful game, you must have a firm grasp of its currency flow. Gauge diagrams can be of great help in this regard.

Applicability

The Currency Design Pattern should be used when you want to have one gauge exert mechanical influence over another gauge in your system.

Example Structure

Although it is not always the case, the arrows between various gauges in a gauge diagram usually represent Currency flow. The following gauge diagram represents a significant portion of the combat system of *The Riddle of Steel*, which is considered by many game designers to be well thought-out. All arrows represent currency flow. All exchange rates are 1-to-1.



Consequences

Currency ties gauges together and quantifies the mechanical influence one gauge has on another.

Implementation Concerns

Some games have non-unitary exchange rates between gauges. For example, subtracting two points from one gauge may allow a player to add only one point to another gauge. This would be a 2-to-1 exchange rate. In general, exchange rates other than 1-to-1 require additional overhead. They should be avoided where possible. That doesn't mean that you should entirely avoid exchange rates other than 1-to-1 all the time. It only means that you should always have a good reason for doing so.

Let's say that we've designed a game that contains gauges A, B, and C. Further, let's suppose that mechanical influence may be transferred from A to B and from B to C. So, mechanical influence (Currency) may be transferred from A to C through B. Suppose the exchange rate from A to B is 2-to-1 and the exchange rate of B to C is 1-to-2. Transferring points from A to B to C demands a division and a multiplication by 2. Logically, a division and a multiplication of the same number should cancel each other out. But, what about rounding? Is it possible to transfer a single point from A to C if I have to divide by 2 first, resulting in 0.5 points being added to B? Perhaps, but maybe not, depending on your design goals. Even in this trivial example, a non-unitary

exchange rate demands additional mathematical overhead and raises questions that could have been avoided by a uniform 1-to-1 exchange rate.

Since game designers generally strive for simplicity and clarity in their creations, it behooves us to avoid the complications of exchange rates. In other words, we would like for a point of A to equal a point of B to equal a point of C. Simple 1-to-1 exchange rates allow the easy flow of currency between gauges. Whenever we can lower a game's handling costs without any degradation to game capability, we have a win/win situation that should not be overlooked. So, all paths allowing currency flow throughout a game subsystem should use the same scale of currency unless there is some truly compelling reason for not doing so. If a game demands more than one type of currency, the subsystems using the different currencies should be entirely isolated from one another. Think twice before giving up on a 1-to-1 exchange rate. And then, think again.

Samples

We could design a game to have 6 character attributes whose values are set by having players spend a pool of points intended for that purposes (see the Point Spend Attribute pattern). Let's give them 12 Attribute Points to spend altogether. These attributes will feed directly into contests. We will resolve contests by rolling a d20, adding in pertinent attribute values, and comparing the sum to a threshold. To continue the currency flow, we'll apply the difference between the rolled sum and the threshold to subsequent rolls, such as damage. So, a really good hit would deliver really good damage regardless of the weapon used. That way, currency would flow all the way from spending Attribute Points through conflict resolution on a 1-to-1 exchange rate.

Known Uses

Sorcerer character creation starts with a resource of Attribute Points which players spend on the Attributes of "Stamina," "Will," and "Lore." Once these are set, Humanity is set to the maximum of Stamina and Will. In contests, the pertinent attributes are used to add dice to a dice pool on a 1 point = 1 die basis. Contests result in degrees of successes (counted in numbers of successes) and those successes can be applied to subsequent rolls on a 1 success = 1 die basis. So, currency flows from initial character creation by spending Attribute Points all the way through to conflict resolution using a 1-to-1 exchange rate.

Universalis currency is actually given the name "Coins." Coins are used to purchase facts, such as character traits, which are ranked on a 1 coin = 1 rank basis. These previously established facts can be directly applied as dice in contests. Coins may also be spent in contests by any player to augment either side's dice pool on a 1 coin = 1 die basis. The total number of coins expended on a character sets the character's "Importance" rating. To kill a character or otherwise remove it entirely from the game, a number of coins equal to the character's Importance rating must be spent. So, currency is front and center in Universalis and is even passed around as physical objects (coins). It flows from the player's Coins pool, which is refreshed at the beginning of every scene and augmented by introducing conflicts. Coins purchase facts, and the number of coins spent on those facts flows on through conflict resolution. All exchanges are 1-to-1.

Gauge

Intent

Provide a concrete way for an abstract concept to have mechanical interactions with a game system.

Also Known As

Not applicable

Related Patterns

Attribute, Currency, Resource, Skill, Trait

Motivation

Understanding gauges is vital to designing mechanical game systems. A Gauge is a graduated value generally associated with a name. Commonly, the graduated values are numbers, but this is not always the case. Gauges are introduced into role-playing games in order to highlight some key game concept that the designer wants to affect and/or be affected by other key game concepts. Gauges are the building blocks of any mechanical game system. It is the relationships that a gauge has with other gauges that gives it its true meaning within the system. No gauge has any purpose independent of its connections to other game entities. In short, no gauge stands alone. What purpose would Hit Points have independent of Damage? What value would a Resource have without some Consumer on which to spend it? Why possess an Attribute if it affects nothing?

Applicability

Along with Currency, the Gauge design pattern is probably the most widely used pattern of any RPG Design Pattern. Quite often, it is overused. Introducing a Gauge into your game makes a statement about what you, as the game designer, feel is important to your game. Using too many of them dilutes the impact that any single gauge has on your system and thereby detracts from the game's central concepts. Use a Gauge when you

1) Want to want to emphasize an important game concept,
2) Want that concept to play a mechanical role in your game,
3) Have seriously considered not representing that game concept as a gauge,
4) Having properly pondered, concluded that introducing a gauge brings more of a focus on the game's core rather than distracts from it.

Example Structure

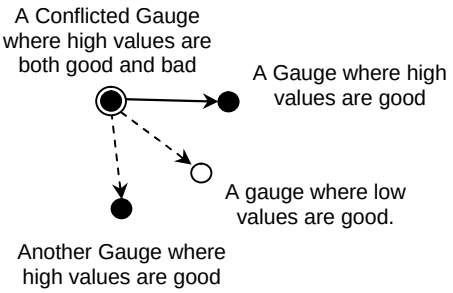


Figure A

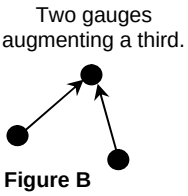


Figure B

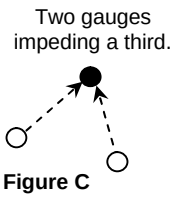


Figure C

Consequences

A Gauge introduces a metered value for what would otherwise be an abstract concept. It thereby gives a concrete form to that abstraction, allowing it to manipulate and be manipulated by a mechanical game system. A gauge is an artifice, a simulacrum of an underlying idea whose purpose is to provide a hook into a game system. It is nothing more or less.

Because of their artificial natures, gauges have the potential to distract players from their immersion in the game world. The more gauges your game contains, the greater the danger of this happening. So, use them where appropriate, but keep them to a minimum. The actual number you will use, of course, depends on your design goals. High-fidelity games purposely incorporating lots of metaphorical knobs and dials to tweak in order to satisfy gamers seeking lots of "crunch" will naturally tend to have more gauges than lighter-weight games intended to "fly under the radar" as much as possible.

Implementation Concerns

As stated before, gauges have meaning only in relation to each other. Most games give gauges numerical values and impose arithmetic relationships between them. That is, one gauge value is added to or subtracted from another gauge value to generate a third gauge value. Diagrams of these kinds of gauge relationship can be found above in Figures B and C. But, do not think that addition and subtraction are the only ways gauges can affect one another: you do not need to limit yourself in this way. Consider these alternatives:

1) One gauge value could be set as the maximum or minimum of two or more other gauges.
2) Along the same lines, one gauge value can act as a ceiling or floor to another gauge value.
3) One gauge value can act as a threshold to another gauge value.
4) One gauge could "spill over" into another gauge value when it "fills up" past a certain value. So, one gauge has a maximum value and anything more added to it cascades into another gauge.
5) Gauge values can be divided or multiplied together or one can be raised "to the power" of another. (It's a good idea to stay away from these kinds of combinations for gauges that are calculated "on the fly" during play, though.)

Samples

A trait-based game might have players select "Verb" traits and "Noun" traits for their characters and give them ranks by spending an allotment of "Trait Points" given to them for this purpose. Players have their characters perform actions based on these traits, but cannot use any single trait in isolation. Rather, Verbs and Nouns must be used together and the rank of the Verb/Noun pair equals the minimum of the ranks of the Verb and Noun. So, a player that was playing a baseball player in a game exploring the effects of money and drugs on professional sports might give his character a Verb trait of "Resist" and a Noun trait of "Steroids" with ranks of 2 and 3 respectively. The "Resist/Steroids" trait pair rank would then be the minimum of the two ranks, or 2.

Known Uses

The Game Summaries section overflows with gauges applied to various purposes. In fact, every single studied game contains gauges of one form or another. If you want to explore a minimalist extreme in this regard, take a look at the summary of Puppetland.

Rank

Intent
Allow players to customize their characters by assigning individualized proficiency ratings to abilities.

Also Known As
Attribute Level, Skill Level

Related Patterns
Skill, Skill Tree, Template, Trait

Motivation
The Rank pattern individually associates a proficiency rating to all (or some) of a character's abilities. The ratings are usually numeric in nature, but the pattern does not demand this. For example, ratings of "novice," "competent," "professional," and "expert" could be applied instead.

In games using the Skill or Skill Tree patterns, character abilities are selected from pre-defined lists of abilities. Those patterns assume that characters will gain proficiency in their skills as play progresses. Games using the Traits pattern may also decide that allowing varying degrees of proficiency in abilities is appropriate. However, none of these patterns specifies the granularity of player control over his character's abilities. Some games uniformly raise the proficiency of all (or many) abilities based on a single character rating (see the Level pattern). Others allow ability rankings to be customized on an ability-by-ability basis. Games allowing individual ability ratings to be controlled by the player exhibit the Rank pattern.

The primary reason to provide this fine a degree of control is to give players tools to better achieve their visions of their characters' personas. Ability ranking systems often force players to trade off their characters' proficiency rankings in some abilities to increase them in others. Since no ability rating can be raised without some cost to other abilities, a player must determine his character's priorities. From the choices that are made, the player's aspirations are, to some degree, revealed.

For some players, a primary motive for favoring a ranking system is to attain better "realism" than they could if they played a game lacking this feature. After all, real people possess skills at various levels of ability, so why shouldn't characters? Such people often find systems that use a uniform "level" for all abilities to be too confining.

Applicability

Use the Rank pattern if you

1) Are using the Trait, Skill, or Skill Tree patterns in your game design.
2) Want to provide players a great deal of control over ability proficiencies.
3) Are willing to accept the bookkeeping overhead needed for players to individualize their character's ability rankings.

The Rank pattern demands additional accounting work from the players, since every ability must somehow be associated with a ranking. If you do not want to require the additional overhead demanded by the Rank pattern, you might want to consider the Level pattern instead.

Consequences

The Rank pattern gives players a good amount of input into their character designs. It forces the player to make trade-offs between the rankings of his character's various abilities, allowing him to gain greater proficiency in one or more abilities at the expense of others. The player's priorities are shown through his decisions. Allowing players to customize their characters to the extent allowed in the Rank pattern does have its costs, though. Additional bookwork of keeping track of a ranking for each skill is required at the very least.

Implementation Concerns

When you use the Rank pattern, your key decision will lie in how to trade off the proficiencies of one ability for those of another. Some games using this pattern make it more and more expensive to raise a particular ability's rank as it increases, but this is not a requirement of the pattern. If you use a fixed cost scheme to raise ranks, you will either have to decide on some way to limit the amount a character can spend in a skill or have some viable means of dealing with very large rank values. Of course, you could just make all ranks extremely expensive so that a player can only raise a few ranks at most, but that would conflict with the primary reason to use the Rank pattern in the first place: to allow players to customize their characters.

Of course, any kind of "cost" to raise a rank implies that a player has some resource from which to draw to increase his character's rankings (see the Resource pattern). Many games use a resource tied to how "seasoned" the character is, or how much "experience" he has accumulated by overcoming obstacles. However, there is no reason you must tie yourself to an experience based resource to control ability ranks, should you decide some other measure is appropriate for your game. Nor should you necessarily constrain your game to the traditional mode of having players raise character ranks by drawing from resources under their control. It is not too hard to envision a game whose focus is on inter-player negotiation that prohibits players from spending their own resources on their own characters' ability ranks. Such a game might require players to negotiate for other players' resources to increase their own characters' skill and trait ranks.

Assuming you have players assign ranks by drawing from some resource, you need to think carefully about what ability ranks can be assigned from a given resource. For example, suppose you design a game with the following features:

1) Characters possess a fixed number of attributes and a variable number of skills.
2) It uses a classic "attribute value + skill rank + die roll" resolution system.
3) Any given attribute applies to a broader range of situations than any given skill.
4) Players spend "Experience Points" to gain skill ranks.

Let's suppose you also decide to allow players to spend "Experience Points" on gaining attribute ranks. By doing so, you may have just created a subtle problem. Attributes are used in a wider variety of situations than skills. So, attribute ranks are more valuable than skill ranks. It behooves players to spend more of their resources on raising their attribute ranks than in raising their skill ranks. The degree to which this is true depends on the disparity between the number of attributes and the number of skills. Essentially, you have created an exchange rate between attributes and skills which favors attributes over skills. A shrewd player will favor his character's attributes at the sacrifice of skills. If you are not very careful in designing your ranking system, skills may inadvertently become completely irrelevant. One of the best solutions to this dilemma is to avoid it entirely: associate one resource with attribute ranks and an entirely independent resource with skill ranks. Do not allow *any* form of exchange or interaction between these two resource pools. Note that the same argument holds regardless of the kinds of gauges to which ranks are applied (Traits, Attributes, Skills, Handicaps, etc). If the ranks of one kind of gauge are more valuable than those of another, you need to keep their resource pools separate.

If you plan on giving characters a large number of ranked abilities, you have another concern. That is how you are going to balance flexibility in your system with the bookkeeping overhead it requires. Do you assume all skills are equally hard to learn (and gain ranks) or do you make some skills easier to learn than others? Making the cost structure uniform for all skills would lower the complexity of your system (and reduce bookkeeping), but takes away a powerful design tool you could use to balance one skill against another. For example, in many fantasy games, big swords and axes are often preferred over daggers and other small weapons. This is because big weapons presumably deliver more damage due to their "superior" size. But, if you allow characters to gain proficiency in small weapons at a fraction of the cost of big weapons, you can make weapon choice an arbitrary decision. Players using such a system are much more likely to choose their weapons based on character concept and other aesthetic reasons rather than from concerns about effectiveness.

Be aware that the Rank pattern's bookkeeping overhead can quickly balloon out of control by poor design decisions with very little tangible benefit to the game overall. This can happen if you decide to make the costs of gaining a rank variable depending on circumstances. You could decide that the cost of gaining a rank depends on a die roll. If a player is "lucky," he suffers only half the cost than if he is "unlucky." Alternately,

you could make the first rank gained within a "game month" to cost less than the second, third, or fourth ranks gained in the same month. While the incremental costs of these kinds of rules seem low at first, the cumulative costs can become prohibitive and painfully obvious in other circumstances. For example, suppose a player needs to work up a powerful character from scratch. What are his character's costs going to be for gaining all of his various skill ranks? If the costs to gain a rank are dependent on factors other than skill difficulty and rank, he will have to devote time to rolling dice or performing table lookups individually on *every* skill rank his character gains. Gaining a rank of 10 on a single ability would then require 10 separate calculations! If, on the other hand, your game's rank costs depend only on difficulty of an ability and the desired rank, then a single calculation can be performed for every ability to determine the costs.

Samples

Melissa is working up a new character for a cartoon-based game. She has decided to create a clumsy dim-witted dog named "Caper" that has turned to a life of crime as a cat burglar for the simple reason that he wants to steal cats. However, Melissa wants Caper to be particularly bad at his job. She wants him to accidentally make noises at inconvenient times, by stepping on squeaky floorboards or knocking over vases, and have a habit of breaking things, the more valuable the better. Melissa wants Caper to have some useful abilities too. She wants Caper to be able to put on a skimpy black mask to hide his identity in a "if-I-put-on-my-glasses-you-can't-tell-I'm-Superman" kind of way. Caper will also have a Cat Burglar's Almanac that he carries around in a burlap bag from which he gets all of his "brilliant" ideas. Finally, she wants Caper to be able to climb through upper-story windows through the use of a grapnel hook and rope that he also keeps in his bag. After all, what good is a cat burglar if he can't throw a grapnel through a second-story window to the sound of breaking glass? The game Melissa is playing allows her to assign both positive and negative ranks to skills, with negative ranks being as expensive as positive ranks, but making a character atrociously (and, hopefully, humorously) un-adept at the skill. (The game gives rewards based solely on whether you make other players laugh : 1 Character Point per chuckle.) The cost to gain a given rank in a skill equals the skill rank value multiplied by itself. To start, Melissa has 50 "Character Points" to spend on skills and decides to use them in the following way:

> Stealth: -4 (costing 16 "Character Points")
> Reading Comprehension: -2 (costing 4 "Character Points")
> Disguise: 5 (costing 25 "Character Points")
> Climbing: 2 (costing 4 "Character Points")
> Picking Locks: 1 (costing 1 "Character Point")

Known Uses

Dungeons & Dragons v.3.5 has a list of 45 basic skills to which all characters have access (see the Skill pattern). Players spend "skill points" to raise their ranks in their skills. The number of skill points a player has to spend is determined by both his character's class (see the Class pattern) and level (see the Level pattern). "In class" skills can be raised to a maximum of the character's level + 3 at a cost of 1 rank per skill point. "Cross class" skills can be raised to a maximum of only half the "in class" maximum at a cost of 2 skill points per rank.

GURPS (Basic Set) has a list of approximately 174 skills, depending on how they are counted (see the Skill pattern). "Bonus points" can be spent on raising skill "levels," which actually corresponds to the Rank pattern (rather than the Level pattern). Skills are segmented into Mental and Physical categories and are rated as "Easy," "Average," "Hard," or "Very Hard." The cost to raise a rank is dependant on the difficulty of the skill and the character's attributes of DX (for Physical skills) and IQ (for Mental skills).

HeroQuest has "abilities" that are essentially traits (see the Traits pattern). Very broad abilities are known as "keywords" that act as templates (see the Template pattern), and represent all abilities common for an occupation, culture, religion, and the like. Each ability has a "rating" that is essentially a number between 1 and 80, but is represented by four distinct levels of "mastery," each of which is subdivided into values of 1 to 20. So, a rating of 44 would be represented as 4 at mastery level 2 (20 x 2 + 4 = 44). In conflict resolution, a contestant with a higher mastery level in the pertinent abilities has a distinct advantage over those of lower mastery level.

The Pool uses "traits" to describe character abilities (see the Trait pattern). In this game, a trait is associated with a numerical value that allows a player to add dice to his character's "Dice Pool" (see the Dice Pool design pattern) in resolving conflicts involving the trait. The more dice used in resolving a conflict, the more likely a character will succeed. The cost of increasing the "bonus" for a trait is exponential, though, and this cost is exacted from the same pool that feeds dice to all conflicts in which the character participates (see the Currency pattern). So, there is tension between building up a character's long-term Traits and his likelihood for short-term success in all endeavors. Success in any conflict results in the player choosing between adding a die to his pool or controlling the outcome of the attempted action. Failure indicates that the character loses whatever dice were "gambled" on the attempt (see the Resource pattern).

Rolemaster Fantasy Role Playing has 158 skills segmented into 47 skill categories (see the Skill pattern). The game gives players "development points" to spend on improving their character's skills, which are awarded when the character gains levels (see the Level pattern). The cost to raise a skill is uniform across all skills within a category. The actual costs also depend on the character's class (see the Class pattern) and are variable depending on whether it is the first, second, or third skill rank gained in the skill at the current level. Depending on the skill and class, a character may be

completely unable to raise a rank in a skill, or may be able to raise only one or two ranks per level. For example, a Bard character has a cost of 2/7 in the Subterfuge : Stealth skill category. This means that when he gains a level, the Bard can raise any Subterfuge : Stealth skill (Hiding, Picking Pockets, Stalking, Trickery) by one rank by expending 2 DP's (development points). He can raise the same skill another rank by spending 7 DP's. He cannot raise the same skill any more ranks until gaining another level.

Universalis allows players to apply "traits" to characters through the expenditure of "Coins," such as "Loves Children" or "Blacksmith" (see the Trait pattern). Even a character's name is a trait in Universalis that must be purchased through coin expenditures. The more coins that are spent on a particular trait, the greater its "importance" rating. The overall "importance" of a character to the story is the sum of all of the "importance" ratings of each of his individual traits. To "kill" a character or otherwise entirely remove him from play, a number of coins must be spent equal to his overall importance. Note that this does not work as a form of "Hit Points" (see the Hit Points pattern for details), because a wound caused to a character by one player can actually allow a player to add the trait "Wounded on Left Thigh +2" to the character by spending 2 coins, which actually makes it *harder* to remove the character from play.

Resource

Intent
Provide a limited quantity of game-world influence that a player may manage, expend, and earn in pre-specified ways.

Also Known As
Not applicable

Related Patterns
Attribute, Gauge, Trait

Motivation
Game designers often wish to allow players to customize their characters to limited (but sometimes variable) degrees and within certain boundaries. This forces the player to make conscious choices on what is important to his character. A resource is a gauge that is usually associated with a numerical value. The value can increase as players "earn" more of the resource and it can decrease as players "spend" it.

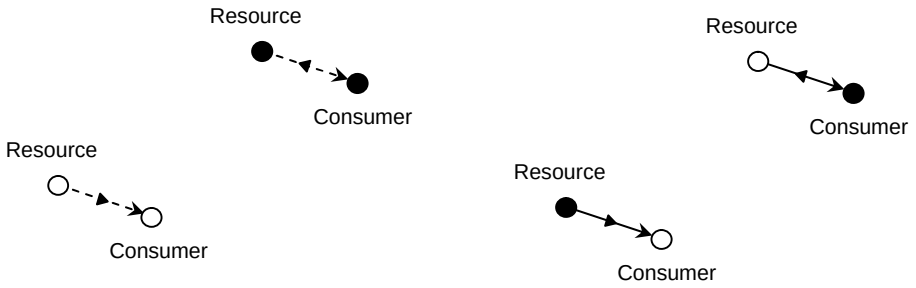
An example of this is a Wild-West style game having a "money" resource that gives new characters $10 to purchase equipment. Depending on the game world economy, this might allow a player to purchase a horse or a six-shooter, but not both. On the other hand, his $10 can in no way allow the player to, say, increase his character's Intelligence attribute. (This assumes, of course, that this Wild-West style game sticks to a reasonably historic American West setting. A bizarre Wild-West/DUNE-style game blend could be created where the purchase of Spice would temporarily give a character "Mentat" abilities, and thereby raise his Intelligence rating.)

What distinguishes a resource from other gauges is that a resource can be "spent" by conscious choices of the player. Some resources can be raised by a player's choice as well, while others are raised when certain events occur.

The altering of a resource value does not have to be automatic. A game resource can be set up in such a way that every time the resource is used, the player gambles on the resource value lowering. Similarly, a player may decide to perform some action on the mere chance that his resource value will increase as a consequence.

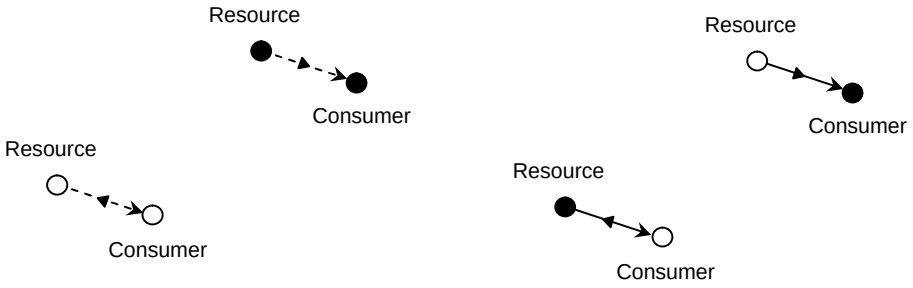
Example Structure

The Resource design pattern can be easily recognized in a gauge diagram. It consists of one gauge acting as the resource and at least one other gauge acting as the consumer. Between each resource and consumer, there is a relationship from the resource to the consumer. The nature of the relationships (direct or inverse) depends on the whether players want large or small values associated with the resource and consumer gauges.



For example, if players want both the resource and the consumer to have large values, the circles representing the gauges are filled in. The relationship from the resource to the consumer is an inverse relationship (dashed arrow) because spending the resource from a high value to a lower value tends to produce larger consumer values. The relationships are also adorned with triangles. These adornments usually indicate that accumulating more of the resource does not hinder the consumer, but spending the resource aids the consumer.

Note that spending something you want in order to buy something else you want is not the only possibility, though. It is also quite possible to design a resource/consumer relationship that allows a player to spend something he doesn't want in order to buy something else he doesn't want. For example, you could incorporate a Trauma Gauge in your game that also acted as a Resource. You could allow players to spend the Trauma Gauge points to "buy" Wound Traits. Players would be willing to do this in cases where a high Trauma Gauge value hindered a character more than the purchased Wound Traits. The gauge diagrams for these kinds of resource/consumer relationships would look like the following:



Applicability

Use the Resource pattern when you want to force players to make important conscious choices in their characters' make-ups or actions.

Consequences

The primary benefit of the Resource pattern is that it emphasizes important game concepts by bringing those concepts to the forefront in the players' minds. It forces players to pay attention to the resource and balance its value with other concerns. Because of this, you probably want to focus your game design efforts on a relatively few types of resources. The more resources you add to your game, the more complex it will become from the players' viewpoints and the greater amount of bookkeeping will be involved in its maintenance.

Implementation Concerns

If you introduce a resource into your game, you must clearly specify to what the resource can and cannot be applied. Otherwise, the ambiguity is likely to result in unnecessary debates among players during game play.

If you have more than one resource in your game, it is advisable to keep the resources isolated from one another. That is, don't allow one resource to be spent to gain a second resource and then allow that second resource to be spent to gain the first resource without some kind of other restriction on how and/or when the resources can be spent. Otherwise, what you really have is one resource with an overly complicated implementation. This is especially true if players can gain an advantage simply by exchanging one resource for another and back again. For example, suppose a player could gain 2 "Character Points" by spending one "Fate Point." And, he can gain 1 "Fate Point" by spending 1 "Character Point." By simply alternately spending Fate and Character points, a player can gain a virtually infinite supply of both. Any rule that allows this kind of thing is a design flaw. No matter how obscure the rule, be assured that any such flaw will be identified and exploited by your players to the detriment of game play.

Samples

Wanting to give his players a little more control over the events in his game, James has created a house rule for his campaign. Every session James gives his players one "hero point." One "hero point" may be used to re-roll any contest as desired by the player.

A game that awards "Experience Points" to characters for overcoming hardships can use experience as a resource. Doing so requires allowing players to "spend" the points in some way, such as in raising their characters' "ranks" in their skills.

Known Uses

The Pool is a game centered on a resource called, oddly enough, "the pool." This resource is a number of dice from which the character can draw (and gamble) in resolving conflicts. However, if a player loses a conflict, the dice he gambled from his pool are permanently lost. If he wins the conflict, though, he has the option of either adding one die to his pool or narrating the outcome of the conflict (which gives him a substantial amount of control over game events).

TORG has a numeric resource called "Possibility" around which the game is heavily centered. Starting characters get a certain amount of Possibility points to spend on skills. The Game Master awards more as play progresses based on the obstacles players overcome (zero to 3 per act and 6 to 12 at the end of an adventure). Other than raising skills, possibility points can also be spent to augment die rolls, to counter an opponent's expenditure of possibility points, to improve attributes, to reduce damage, and other effects.

Miscellaneous Patterns

Alignment

Intent

Differentiate characters by segregating them into different categories that define how in-game events affect them physically. Characters can also be distinguished by limiting their abilities based on these same categories.

Also Known As

Not applicable.

Related Patterns

Faction, Idiom

Motivation

The goals of Alignment are easy to misconstrue. This is because many games present alignment as a common characteristic that specifies how a player should portray his character. Alignments are usually specified with connotive words rather than numbers, the most common of which are "Good" and "Evil." To extend the field of aligned behaviors to a wider range of possibilities, many games specify a number of alignment characteristics, each of which must be assigned values. In addition to "Good" and "Evil," a game might require a player to decide between "Lawful" and "Chaotic" or "Social" and "Antisocial," etc.

Because of these moralistic names, it is easy to come to the conclusion that a game having alignments is actually trying to persuade players to portray their characters in certain ways. The text might even say this. However, the Alignment pattern does nothing to promote role-play in any mechanical way (such as by rewarding players for doing so). Thus, the pattern cannot really be described as a mechanical means of promoting role-play. (Note that the Faction pattern, which is similar to the Alignment pattern, *does* provide rewards for role-playing characters according to specified belief systems. Alignment and Faction are often used together, so it is easy to confuse the two.)

The Alignment pattern *is* useful, though. One must simply recognize that the actual design goal which Alignment satisfies has nothing to do with promoting role-play. Rather, its purpose is to differentiate characters by assigning various physical effects to some in-game events based on alignment categories. It can also be used to distinguish characters by constraining character abilities based on their alignment category. A character's alignment might therefore limit the character to a subset of a game's career choices. For example, a player wishing to play a "White Witch" might be required to select a Good alignment. Selecting this option might simultaneously preclude the

character from ever becoming a "Black Witch." A game might even view Good and Evil as physical properties that can be detected and manipulated. Thus, a "White Witch" might have specialized skills that have different effects based on the target's alignment. She might get a palpable sensation whenever evil approached, for example. Or, she might be able to summon a "Radiance of Goodness" to aid her Good companions, hinder her Evil foes, or both.

Applicability

As a role-playing aid that gives guidance to players concerning the manner in which they should portray their characters, the Alignment pattern does a poor job. Other patterns, such as the Faction and Idiom patterns satisfy this goal to a far better degree. It is highly recommended that you understand these patterns before deciding to use the Alignment pattern as a role-playing guide.

As a means of differentiating characters based on pre-specified categories, the Alignment pattern excels. Use the Alignment pattern if your goals include:

1) A desire to define a fixed set of broad categories into which each character is placed.
2) A desire to have in-game effects vary from one character to another based on his assigned category.
3) A desire to limit player options based on the category to which a character is assigned.

Note that goals 1 and 3 can be satisfied by the Class design pattern. If you do not want to vary the in-game effects of character actions based on a character's alignment, you might want to consider that pattern instead.

The Alignment pattern tends to work well with the Skill and Gift patterns, but is less harmonious with the Traits pattern. The reason is simple. If you want to vary the in-game effects of various actions based on alignment, you need to specify exactly *how* those effects vary. Pre-defined skills and gifts provide this opportunity in that each requires its own description. The traits pattern, on the other hand, demands a more general rule describing how each player-specified trait interacts with the various alignments. It is telling that none of the games analyzed in researching this book used both the Trait and Alignment patterns together.

The Alignment pattern mimics the Faction pattern in structure, in that both require characters to be placed in groups. Consequently, many games combine the Alignment and Faction patterns. Player options are constrained by a character's alignment, in-game effects vary based on the alignment, and the alignment serves as a faction promoting conflict between the different categories.

Consequences

The Alignment pattern essentially adds a characteristic to each character that interacts with the game-world reality as if it were a physical property. It can often be detected,

leveraged, and manipulated as in various ways by game rules specifically designed to do so. Although alignments are often identified by moralistic words such as "Good," "Lawful," "Evil," "Antisocial," and the like, they do not provide any mechanical effect to encourage players to role-play in any particular way (although many players will do so anyway because they closely associate a character's alignment with his behavior patterns).

Since the alignment pattern seeks to vary the effects of actions based on a character's alignment category, it can add a large burden to the game-writer's shoulders. The set of possible alignments essentially spans the entire game and has far-reaching consequences. It is likely that a large portion of the skills and gifts contained within a game's text will have alignment-based effects. Each of these effects requires its own discussion to clarify the differences. This means that the various skill and gift descriptions will be lengthier than if no variable effects existed. You might decide to lessen your workload by having relatively few skills with alignment-based effects. However, if you do so, you probably should reconsider using the Alignment pattern. After all, why complicate your game for something that will only have a minor impact?

Implementation Concerns

If you decide to use the Alignment pattern, you need to concern yourself with categories to which you are going to assign characters. The whole "Good" versus "Evil" alignment concept has been explored by a great many games. So, you may want to avoid using these alignment categories in your own games to differentiate it from its predecessors.

Note that a game exploring the dangers and moral dilemmas faced by mountain climbers might categorize characters as "High Altitude Acclimated," "Moderate Altitude Acclimated," and "Sea-Level Acclimated." After all, these are broad categories that limit player options and the specific category would define the effects that a high-altitude environment would have on a particular character. By spending sufficient time at a given altitude, a character may gradually change from one category to another. So, it satisfies the pattern in a way that completely strips out all moralistic judgment of character behavior.

Samples

A game with an alignment system might segment alignments into two aspects: Goodness and Sociability. These would be set as Good or Evil and Social or Antisocial. Such a system might provide the following definitions:

> **Social**
> A Social character befriends others through his trustworthy acts. He helps any other character in desperate need if possible. Social characters also expect others to aid them in their needful times.

Antisocial
An Antisocial character uses other party members to suit his needs. He quickly picks fights with those standing in his way. Of course, he may act highly social as long as it serves his needs.

Good
A Good character has mercy on those who ask and deserve it. He serves justice and demonstrates kindness to all he meets. Good characters defend townships from evil invasions. They save fair princesses from evil wizards. A good character would attempt to slay any slavering, vicious, hungry ogre threatening a nearby orphanage. Conversely, a Good-aligned character more easily gets help when needed. Defending a town from an angry ogre endears a character to those townsfolk saved.

Evil
An Evil character delights in the misery of others. He strives for personal power and allows no sense of mercy or justice to interfere with gaining it. Glory, pleasure, power, and wealth are the major aims of an Evil character but his methods may seem perfectly innocent on the surface.

Such a game might then restrict characters to certain classes (see the Class pattern) based on their alignments. For example:

Cult Leader

A Cult Leader is the head of a religious sect. Although some cult members may join the leader on their own volition, most are coerced through brainwashing. Some cult leaders are actual priests of evil deities while others are just greedy bastards trying to make a buck. Interestingly enough, a widespread cult may have many leaders which were "promoted" from the brainwashed cult members. Even as these cult leaders brainwash others, they fully believe in what they preach.

Alignment Restrictions
To become a Cult Leader, a character must be *Antisocial* and *Evil*.

Gifts
Attracting Followers, Beguiling, Imitating Voices, Inciting Riots, Quoting Religious Phrases, Sleight of Hand, Throwing Voice

Skills
Brainwashing (+4), Fast Talking (+3), Interrogating (+2), Inspiring Loyalty (+2), Raising Morale (+2), Manufacturing Hallucinogenic Poisons (+1), Torturing (+1)

The game might also include various skills (see the Skill pattern) that have varying effects based on alignment. For example, it might contain a magical spell such as the following:

Shooting Star

Alignment Restrictions: The caster must have a *Good* alignment.
Affected Area: One creature
Casting Time: 10 seconds
Duration: Instantaneous
Range: 100 feet

Shooting Star creates the spectacular sight of a sparkling ball shooting toward the caster's target. To strike the target, the caster must make an Attack Roll with Range Weapon Adjustments. The ball is quite harmless to any Good aligned creature. To Evil creatures, however, it represents wrathful vengeance. Any Evil creature struck by a Shooting Star sustains 1d8 fire damage per spell level.

Note that the text describing the various alignment categories seems to indicate that characters should be role-played in certain ways. However, the alignments are only used to identify what career options are available to a character and/or how alignment-based influences affect him.

Known Uses

Dungeons & Dragons v.3.5 has an alignment system with two aspects, each of which can be set to one of three values by the player. The first aspect has the options of Lawful, Neutral, and Chaotic while the second aspect has the possibilities of Good, Neutral, and Evil. Alignment is used as a prerequisite to attaining certain classes (see the Class pattern), has skills whose effects vary based on alignment, and serves as a general role-playing guideline for players.

RIFTS has three alignment categories of Good, Selfish, and Evil, one of which the player must choose for his character. In each category there are sub-types. From the Good category a player may choose Principled or Scrupulous. From the Selfish category, he may select Unprincipled, or Anarchist. From the Evil category, he has the options of Miscreant, Aberrant, and Diabolic. The alignment system has no effect on the game other than to serve as a general guide on what kinds of actions are appropriate when role-playing a character.

Rolemaster Fantasy Role Playing has 39 different "Personality Traits," 20 different "Motivation Traits," and 12 different "Alignment Traits." Players roll randomly to determine which Personality, Motivation, and Alignment characteristics their characters possess. They then make another roll to determine the extent to which that aspect applies. For example, one of the "Alignment Traits" is the Good…Evil characteristic. A percentile roll then determines exactly where on that scale the character lies. Other than as a general guide for players on how to portray their characters, the alignment system does not have any apparent impact on the game.

Warhammer Fantasy Role Play has five alignments, one of which must be chosen for every character. These are Lawful, Good, Neutral, Evil, and Chaotic. Some career

options are only available to characters with certain alignments. There are no apparent differences in how various influences affect characters based on alignment, though.

Game Master

Intent

Assign a single player a special role with different responsibilities than other players, commonly including acting as the final authority in disputes, playing NPC's, describing scenes, and any other tasks not otherwise covered by the game rules.

Also Known As

Dungeon Master, Game Referee

Related Patterns

Contest Tree, Endgame, Structured Story

Motivation

The term "Game Master" is somewhat nebulous. It means different things in different games, primarily because the role of the Game Master varies. The Game Master essentially falls into the "everything else" category. That is, if a crucial game aspect is neither specified as a player responsibility nor handled mechanically, then it falls to the Game Master to assume that task. Some traditional Game Master responsibilities are

1) Acting as the final arbiter in disputes between players.
2) Ensuring all players participate.
3) Describing scenes, including preparing any necessary background materials.
4) Creating and playing non-player characters.
5) Creating challenges for players to overcome when appropriate.
6) Creating a sense of rising tension building to some dramatic climax.
7) Creating a sense of mystery and intrigue in a storyline when appropriate.

Any of these responsibilities can be given to other players or replaced with some purely mechanical process. In fact, *all* Game Master responsibilities can be parceled out in one fashion or another, as is evidenced by the fact that some games exist without *any* Game Master at all (see the Universalis Game Summary for an example).

Applicability

You should introduce a Game Master role into your game when you have a compelling design goal that warrants singling out a player to take on special duties. Certainly, the Game Master role helps many games. But, you should *not* include a Game Master in your game simply because you cannot envision how to design a game without one. Currently most games include such a role, so you may want to consider avoiding it simply to differentiate your game from its competition.

Consequences

Having one player assume a Game Master role puts that player on a different footing than other players. It may be no stronger or weaker than other players, but it is unquestionably different. Depending on your design considerations, this may be good or it may be bad. Some people enjoy taking on the Game Master role, but many people do not. From the purely non-scientific and unverified viewpoint of the author, more people seem to fall into the "do not enjoy being Game Master" camp than otherwise. So, mixing a Game Master into your game's design may diminish its potential audience.

Implementation Concerns

As stated above, Game Masters essentially take on any "left-over" tasks that are not handled in some other fashion by the game rules. So, designing a Game Master role boils down to picking his responsibilities. If you don't want him to assume a particular duty, then you need alternatives. For this reason, this description will focus on how to replace many of the traditional Game Master roles with other techniques. If you dream up some new innovative way to tackle a given job, do not feel limited by what follows.

1. **Acting as the final arbiter in disputes between players:** Replace this responsibility with a simple dice roll, coin flip, or bidding process. This will probably be a Negotiated Contest. The technique may match your game's normal conflict mechanic, or it may be entirely different. Note that this represents a form of *player* conflict resolution rather than *character* conflict resolution.
2. **Ensuring all players participate:** You can replace this responsibility with something as simple as a rule that says players take turns going around the table. An even better alternative is to introduce a refreshable resource that players spend to obtain "screen time." Aggressive players burn these resources quickly, leaving timid players with opportunities to contribute. When everyone runs out of the resource, refresh it.
3. **Describing scenes, including preparing any necessary background materials:** Parcel this responsibility out to the players. One way is to provide players with a resource of some sort that they may spend to introduce facts into the game world. In this way, you can eliminate preparation altogether by making world creation part of play.
4. **Creating and playing non-player characters:** Make all characters player characters so that the players play both the protagonists and antagonists of the story. That does not mean that a given player plays both sides (unless that is appropriate for your game), it merely means that antagonists are played by players and protagonists are played by (potentially different) players.
5. **Creating challenges for players to overcome when appropriate:** Distribute this task to all the players. After all, people challenge one another to games of chess, why can't they directly oppose one another in role-playing games? If challenge is important in your game, be sure to reward anyone authoring it. And, avoid offsetting this reward with harsh punishments should the challenger lose. Otherwise, players will hesitate to create challenges and their play will suffer.

6. **Creating a sense of rising tension building to some dramatic climax:** In all the games researched in our study who incorporated a Game Master, GM Fiat handles this task for a story's overarching conflict. There is no reason that players cannot do as well. However, a mechanical means of generating rising tension may help your game. See the Contest Tree pattern for how to do this.
7. **Creating a sense of mystery and intrigue in a storyline when appropriate:** This is a tough one. (Some games lacking Game Masters have been criticized for lacking mystery as well.) If your game is fine without mystery, this may not be an issue at all. If it is, you can parcel this responsibility out to the players, so that each possesses a secret that other players seek to uncover. Or, some mechanical means of generating mystery can be used (such as the mechanics of the board game "Clue," an example of which is presented in the Samples section below).

It is also possible to simply transfer traditional Game Master tasks to individual players. So, one player may be given the responsibility of "Final Arbiter" while another one is assigned the role of "Scene Framer."

Samples

Let's suppose we want to design a Game Master role for "Clue, the Role-Playing Game" based off of the popular board game "Clue." We'll omit as much as possible in our discussion other than the Game Master's responsibilities, because that is more than enough for a simple example. Because we are basing our creation on a gaming icon, players will expect a certain degree of similarity to the board game. So, we want to retain the "feel" of the original mechanics as much as possible. So, we remove all responsibility for generating a mystery from the Game Master, despite the fact that we are creating a mystery game. The Game Master does have other important tasks, though. For one, he plays a lone detective investigating a missing person report. Mr. Body is nowhere to be seen. Everyone else plays a suspect, whose name and persona they make up themselves. At this point, all players know that the missing person has been murdered, and one of the characters is a killer. However, none of the players knows the identity of the killer and none of the innocent characters is certain that a murder has taken place, although they may suspect it. It is assumed that one of the characters (at random) called the detective to report the missing person. But, the call was given with the approval of all involved (since the killer didn't want to draw any suspicion.)

Before play starts, the Game Master decides on a setting, be it an old mansion, a shopping center, or any other building that may have some interesting rooms to explore. Then, he writes down the names of various rooms of the setting on 3x5 cards. Next, he writes the names of all characters on cards. Finally, he writes down the names of various items that could be used as weapons to kill someone. All of the previously mentioned cards are then read to all the players. As they are read, they are placed into separate stacks of "Room" cards, "Weapon" cards, and "Suspect" cards. The players then give general descriptions of their characters, so everyone can get a feel for the

scenario. The Game Master should keep a separate list of all of the items written on the cards for future reference during play.

Then, the three stacks of cards are shuffled (keeping the stacks separate) and one card from each stack is set aside and placed in its own envelope. No one may look at these cards, not even the Game Master. On the outside of the envelopes are written the words "Crime Scene," "Killer," and "Murder Weapon" respectively. These three cards indicate the location of the murder, the perpetrator's identity, and the weapon used.

Next, all players write down a "secret" about their characters on a 3x5 card that, if discovered, would give the character a motive to kill. The secret must include the character's name. Then all of the cards (save those in the envelopes) are shuffled together and distributed to the players, who may look at them but must keep them secret from other players.

During play, the Game Master has the detective wander through the various rooms, which he describes in detail. It is his responsibility to occasionally wander into named rooms (the ones on the cards) and to inject the various potential weapons in the scenes. The players have their characters enter and exit scenes at will. When the characters are "on stage," the detective interacts with them and asks them questions about the various items he encountered and the rooms he searched.

The players are responsible for inserting hard facts into the narration revealing that the elements on the cards he holds in his hand could not be involved in the crime under consideration: "But, that knife *couldn't* have been used to kill anyone. It's actually a stage prop. Look, the blade collapses into the handle when you stab anything. If *I* was going to kill someone, I'd use that poker over by the fireplace."; "But, Ms. Sampson couldn't have done anything like what you're suggesting. Mr. Body may have discovered she is a cross-dresser, but I know for a fact that she, or, rather, *he*, is a devout pacifist. See, here's his membership card to 'Pacifists Anonymous.'" Similarly, each player is obligated to reveal no hard facts about any elements not listed on the cards he holds. However, he is free to have his character accuse the other suspects of any number of crimes. He can even suggest possible scenarios for how they went about it, even if those scenarios incorporate elements he does not hold in his hand. He may also make up any number of non-verifiable alibis to exonerate his character.

The Game Master may take notes on what he observes. Nobody else may.

When the detective has gathered enough clues so that the Game Master believes he knows which weapon was used in the murder, he has the detective reveal his conclusion to one or more of the suspects. At this point, the envelope labeled "Murder Weapon" is opened and the contents read aloud for all to hear. Whether right or wrong, Mr. Body's body will be found in the next scene and the cause of death will be revealed. Any player may provide the necessary description: "But, you said Mr. Body was hit over the head with a baseball bat. So, where did that bullet hole in his forehead come from?"

After the weapon has been revealed and the detective has gathered enough clues for the Game Master to surmise the location of the crime, the detective once again reveals his belief. At this point, the Game Master opens the "Crime Scene" envelope and reads aloud its contents for all to hear. Whether right or wrong, a strong clue will reveal the location of the crime in the next scene, which may be provided by any player: "You said Mr. Body was killed while sleeping in his bed. But, look! I found a bullet-hole in the Library."

Finally, when the Game Master infers who committed the murder and why, he has the detective gather all of the suspects into a room. Here, he reveals his brilliant deductions for all to admire in rapt awe. The accused is carted off in chains to jail.

The "Killer" envelope remains sealed. Game Over.

Known Uses

Paranoia xp has a Game Master. The Game Master is fairly traditional, but is the only player who may reveal any knowledge of the game's rules in play. Anyone else inadvertently revealing his knowledge of the rules has his character labeled a "traitor" and executed for possessing information above his Security Classification Level. As always, the character is immediately replaced with an identical clone.

Puppetland has a "Puppetmaster." The Puppetmaster provides narrative description of the setting, plays all non-player characters, and judges conflicts based purely on his personal sense of dramatic impact.

Reward Patterns

Attendance Reward

Intent

Provide players with incentives that reward them for showing up to gaming sessions.

Also Known As

E.P., X.P., EXP, Character Points, Development Points

Related Patterns

Failure Reward, Idiom, Narrative Reward, Success Reward

Motivation

The Attendance Reward pattern is a means of rewarding players for attending gaming sessions. The idea is to encourage player participation and help ensure everyone shows up. A game follows the Attendance Reward pattern if it does the following:

1) It gives a player a reward every session having some noticeable in-game benefit to the player without the player having to do anything to earn the reward other than participating.
2) The reward is a purely "meta-game" construct. That is, it isn't any physical thing, such as money or food.
3) The reward is not something that must be expended to play at all. If it must be spent or play grinds to a halt or otherwise suffers greatly, then it is not actually a reward, but rather a mandatory refreshing of some vital resource needed for smooth game flow.

Applicability

You should consider the Attendance Reward pattern in your game if your design goals include

1) Games that run more than a single session,
2) A desire to maintain a consistent group of players from session to session.

If you want to allow characters to be freely "handed off" to other players when their regular players are unable to attend gaming sessions, you probably have no need for an Attendance Reward.

Also, keep in mind that the ultimate reward for attendance is simply having *fun*. If your game is fun, players will attend. The more consistently fun it is, the more consistently it will draw players. If your game isn't fun, no attendance reward will help. Even so, in games that *are* fun, attendance rewards can provide a little extra nudge.

Consequences

As stated before, the Attendance Reward pattern primarily encourages players to diligently participate in a series of gaming sessions. However, it does this by having a real world event (player attendance) affect in-game facts. Having an in-game effect without a corresponding in-game cause can put off players valuing an accurate world simulation above other concerns.

It is worth mentioning that there is an ongoing debate among behaviorists and psychologists that rewards may actually be a long term *disincentive* to desired behaviors (read *Punished By Rewards: The Trouble with Gold Stars, Incentive Plans, A's, Praise, and Other Bribes* by Alfie Kohn). There is little argument that rewards produce short-term changes in behavior. However, the theory asserts that providing rewards for "good" or "appropriate" behaviors eventually transforms activities that start out "fun" into "work." The rewards demonstrate the control the reward providers have over the reward recipients. This means the recipients are likely to lose interest in the activities, as they begin to view the tasks as obligations rather than as challenges. According to the theory, rewards that focus on individual actions are the most detrimental. This effect is noticeable in some seasoned gamers that tire of the "leveling up" and "random encountering" behaviors that Experience Point based games often encourage.

Alfie Kohn doesn't give any realistic advice on what to do in lieu of rewards, though. He just seems to say that people should be allowed to do whatever interests them, and that they will then excel at those tasks. This all makes sense, of course, except that somebody has to fix the sewers when they break, and it is doubtful that there is an overabundance of people who are just dying to explore the enthralling possibilities of sewer repair.

Implementation Concerns

The Attendance Reward pattern rewards player perseverance. In implementing this pattern, you should avoid diluting that strength with contradictory rules. For example, suppose we decide to provide players with an Attendance Reward in the form of "experience points." Further, suppose we also allow "experience points" to be earned by other means in the game, such as slaying monsters. If we then prompt players to temporarily adopt the characters of absent players and allow those characters to gain "experience points" for slaying monsters, we have significantly lessened the incentive our reward provides in encouraging players to attend sessions. After all, the more monsters the character slays absent the owning player, the less important the actual Attendance Reward becomes.

A better option in this case would be to have separate rewards for attending sessions and for slaying monsters. That way, one reward cannot be directly substituted for another and players will *have* to attend gaming sessions if they want the corresponding reward.

Samples

Suppose we decide that we want our game to allow players to influence the outcome of conflicts that they deem especially important. To do this, we give each player a "Divine Intervention" resource. When a point of this resource is spent, they can declare the outcome of an upcoming conflict, circumventing the usual conflict resolution technique. The only way "Divine Intervention" is earned, though, is by attending a game session. Every session personally attended earns them 1 point.

Known Uses

Hero System 5th Edition awards "experience points" which can be spent in character development. Each session is generally worth at least 1 experience point, but players can earn more by being clever, role-playing well, solving a mystery, etc. The fact that this has a minimum value for every gaming session makes this (at least partially) an attendance reward.

Nobilis awards each player a "Dynasty Point" at the end of every game session, provided the players agree that they all had fun. So, this reward is an attendance reward that has the caveat that it is only awarded if all players enjoyed themselves. Dynasty Points can be converted into other important game resources.

The World of Darkness awards each player between 1 and 5 "experience points" at the end of each gaming session. The amount awarded is largely based on GM Fiat, but at least one point is awarded for "just being there."

Failure Reward

Intent

Decouple player success from character success.

Also Known As

Not Applicable

Related Patterns

Attendance Reward, Idiom, Narrative Reward, Success Reward

Motivation

The Failure Reward pattern awards players for character failure, whatever "failure" means within a game's context. A game follows the Failure Reward pattern if all of the following apply:

1) Character actions or goals may succeed or fail.
2) The chances of character success or failure can be influenced by player decisions.
3) A reward is given to the player if his character fails to achieve a goal or perform an action, but that same reward is withheld if he succeeds in that endeavor. (Note that a *different* reward may be given if the character succeeds without violating this pattern. In that case, the Success Reward pattern is also being used.)

A Failure Reward gives players a reason to want their characters to "lose." For many gamers, this is a totally alien concept. After all, what is the point of striving to lose? The answer can only be understood after one first understands that player success and character success need not be tied together. A player is not a character and a character is not a player. One is a real person; the other is a fictional construct. The character is merely a tool by which a player interacts with a fictional world. In a game whose goal is to attain some victory or win condition by means of manipulating a character in a game world, then character success and player success are, usually, tied together. However, in a game whose goal is primarily to create interesting stories, the player goal (creating an interesting story) is independent of his character's goal (becoming the victor in some conflict).

Stories are enhanced by character failure at some level. Any decent book on fiction writing will tell you that a good story requires a conflict between antagonists and protagonists that are evenly matched. That is, a story has to have some central contest between the villains (antagonist) and heroes (protagonist) and it must be evident that both sides of the conflict have a chance at winning. (See *Writing to Sell* by Scott

Meredith, *Immediate Fiction* by Jerry Cleaver, *The Marshall Plan for Novel Writing* by Evan Marshall, and *How to Write a Damn Good Novel* by James N. Frey.)

Most good stories give the villains the upper-hand at first, allowing the heroes only an apparently slim chance of victory. To prove the worthiness of the villains, the heroes must do their utmost to attain their goals and they must fail in the attempt. (If the heroes attain victory on their first attempt, then the villains are easily viewed as unworthy opponents and the story suffers greatly.) One common pattern in fiction writing is to have the heroes try and fail twice. Then, they try one final time where the antagonists maneuver them into a situation that is apparently hopeless. But, through some clever trick or insight of the heroes, they defeat the villains in the story's climax. The rising tension and buildup to the climax hinges on the initial failure of the heroes. So, if you want to create a game that generates interesting stories, you are best served by designing a system where players can rationally decide to have their characters fail at important times.

Another important use of Failure Rewards is as a game balancing mechanism. Small rewards can be given to characters that lose conflicts so that those characters that consistently lose can be strengthened until they finally start winning.

Applicability

You should consider the Failure Reward pattern in your game if your goals include

1) A desire to disassociate player success from character success (in other words, you want for players to strive to have their characters "fail" at times).
2) You are not averse to including a mechanical means of gauging the value of character success versus the value of character failure.

It would be possible to design a game where players *always* strive for character failure. However, the Failure Reward pattern is most often used in conjunction with the Success Reward pattern. So, if you incorporate Failure Rewards into your game, you should strongly consider using other reward systems as well.

Consequences

A game incorporating Failure Rewards decouples character success from player success. In such a system, players will tend to do what is best for them, regardless of whether it is good for their characters.

A game can give players real choice in their decisions to have their characters win or lose by combining Failure Rewards with Success Rewards. To do this, the rewards for character failure must, at times, be sufficiently compelling for players to take that option. To ensure it is a real choice, the rewards for character success and the rewards for character failure should differ in some important way.

Do not think that a combination of Success and Failure Rewards results in a drab system where character success and failure are meaningless. On the contrary, a well

thought-out system of Success and Failure Rewards can have a significant positive impact on a game's overall design. All it means is that players will try to have their characters win sometimes and lose at others. To be effective, there must be real differences between the rewards given for character success and those given for failure.

If a Failure Reward increases a character's long-term chances of success in future conflicts, then it makes that character more mechanically potent. As long as these rewards are greater for failure than for success, they act as balancing mechanisms that slowly build the capabilities of weak characters until they are competitive with strong ones. This is an extremely powerful technique that can be used to make a game design self-balancing.

Implementation Concerns

If a Failure Reward is used in conjunction with a Success Reward, it is important to differentiate the rewards for character success and failure. The differentiation can be qualitative, quantitative, or both. If the same kind of reward is given for both, then a distinction must be made as to the quantity of reward you are providing at various points throughout the game. So, you may deem it appropriate to reward character success but not failure at times and the reverse at others. If your rewards are both qualitatively and quantitatively identical, then you are actually rewarding all actions, whether they succeed or not. Now, if you're trying to stimulate lots of activity in your game without regard to success or failure, there's nothing wrong with rewarding action in itself. However, doing so does not require any sort of artificial division between Success and Failure rewards.

If your game differentiates the rewards for character success and failure in a purely quantitative way by varying the amount of rewards at different times, then you are not really giving players a choice about whether to have their characters succeed or fail. When rewards for two different options are qualitatively the same and only vary by amount, the logical choice is for players to always go for the biggest reward. If you are designing a competitive game where the goal is for players to attain some "win" condition over other players, then this may be appropriate. If not, then you should make sure your rewards for success and failure have important qualitative differences. One such possibility is to trade off character success "now" for character success "later."

If you do not also include the Success Reward pattern in your game, then your design will encourage players to always seek character failure. While such a game is no doubt possible, it may be depressing or frustrating to play. (Either that, or downright hilarious if the goal is to achieve some sort of comedic outcome.) Of course, if that's what you're shooting for, a pure Failure Reward system might be just the ticket.

Samples

Let's design a reward system for a game that tries to encourage the idea that good stories often have characters fail twice in their attempts to win some overarching conflict and succeed gloriously on their final attempt. To do this, we'll give each player a "Plot Points" resource. Die rolls in our game will not directly determine character

success, but rather determine which players are allowed to narrate the outcome of a conflict. How a player narrates an outcome, whether it results in his character succeeding or failing, is up to him. At the beginning of play, we give all players 5 "Plot Points." Before any conflict, all players gain one more "Plot Point" to add to their pool. To keep things simple, on all conflicts we'll have any player with a character involved in the conflict roll a number of d6 equal the number of "Plot Points" they spend. Any values of 4 or more count as a "pip" and the player with the highest number of "pips" wins the right to narrate the conflict's outcome. Ties are re-rolled.

At the beginning of play, we have the players create their characters and negotiate exactly what it is that the villains have done that imposes hardship on the heroes and what the heroes must do (in general) to make it stop happening. In other words, the players negotiate the goals of both the villains and heroes. Next, we break every story down into three "Acts," and every Act into three "Scenes." Each Act is an attempt by the heroes to attain victory over the villains and every Scene describes some action performed by the heroes in that endeavor. The final scene in every act determines who wins the Act. The results of the first two acts *cannot* result in total victory for either side, but it can result in a major win that put the opponents at an apparently severe disadvantage. The outcome of the third Scene of the third Act determines ultimate victory.

Now, let's design our reward system. Remember, we're trying to design a system where the heroes try and fail twice and then ultimately succeed. So, every time a player wins the right to narrate the outcome of a conflict, we give him the opportunity to *double* the number of Plot Points he spent on that Scene's conflict. To win this substantial reward, though, he must narrate the outcome in the heroes or villains favor in a way that depends on how far the Story has progressed:

Act	Scene	Outcome Required to Win Award
I	1	player choice
	2	player choice
	3	heroes fail
II	1	player choice
	2	player choice
	3	heroes fail
III	1	player choice
	2	player choice
	3	player choice

Since ultimate victory is really only decided in Act III Scene 3, players will do what they can to gain as many Plot Points to spend on that scene's conflict. To do that, they have extreme pressure to make sure the heroes fail in the third Scenes of both Acts I and II. Also note that this follows the Contest Tree Design Pattern as well since the results of individual Scenes provide mechanical input to the overarching conflict.

Known Uses

Capes has two important resources: Debt and Story Tokens. Story Tokens are a standard resource in that having more of them is always beneficial. Debt is actually a Conflicted Resource, in that having some Debt is good for a player, but having too much is bad. The conflict resolution mechanic is rather involved, so we won't go into detail about it here (you can see a complete description in the Game Summaries section). However, one important aspect of the conflict resolution system is that "Debt" can be staked on a conflict to improve a player's chances in winning the conflict. If this is done, the winner converts the Debt that he staked into Story Tokens, which are then given to the conflict's loser. So, character failure is a way in which a player can gain Story Tokens if he finds himself in short supply. Thus, it is a Failure Reward.

Dog in the Vineyard conflicts impose "Fallout" on the loser of the contest. The effects of Fallout include injuries to the character, incidental damage to the character's equipment, and the like. However, it also sometimes rewards the player as well. These include such things as adding points to an Attribute, adding a new Trait to the character, or increasing or decreasing the potency of an existing Trait (player option).

Torg has a "Hero Fails" subplot card which guarantees the hero fails at some critical task in exchange for a Possibility Point reward later in the game. The player determines whether or not to play the subplot card, but when and how it is invoked is up to the Game Master once the card is in play.

Narrative Reward

Intent

Provide players with an incentive to create imaginative narrative descriptions of scenes and character actions.

Also Known As

Not Applicable

Related Patterns

Attendance Reward, Failure Reward, Success Reward

Motivation

The Narrative Reward pattern awards players concrete game advantages for describing character actions and game scenes with novel and interesting prose. Many games encourage players to use inventive phrases and approaches to game situations, but fail to provide any real incentive for doing so. The goal for providing these rewards is to shift narration from the repetitive "I swing my sword…I swing my sword…I swing my sword" mode common to many battle-oriented style games into something more varied and engaging. "I grab a handful of flour from the bag sitting on the chopping block and throw it into my assailant's face…I leap up onto the kitchen counter and snatch one of the large hanging copper pots. I then use it to present the thousand and one ways in which a master chef's culinary tools can be applied. I'll start my demonstration by applying the kettle to the *maitre d's* head."

To follow the Narrative Reward pattern, a game must

1) Detail the conditions under which a reward for entertaining narration can be earned by players.
2) Have that reward provide a clear benefit in play.

Although not required by the pattern, some games tie the magnitude of the reward directly to the quality or emotional impact of the description. So an amusing anecdote that raised a smile from the participating gamers would be rewarded as a valid contribution, but not as much as a knee-slapping, tear inducing quip that would be remembered months or years later.

Applicability

You should consider the Narrative Reward pattern for your game if your goals include

1) A strong emphasis on having players continually provide fresh and colorful narration for story events.

2) A willingness to have subjective judgments concerning narrative quality influence game mechanics in ways that can have very real impacts on story events.

Of course, it would be an unusual role-playing game that actually seeks repetition in narrative description over colorful dialogue. However, conventional wisdom in game design states that all rules of a game should focus on its central point. Anything extra distracts from that core idea and should be omitted. If you feel that flavorful narration is of secondary importance in your design goals, you may want to forego any kind of narrative reward.

Similarly, if you feel that subjective judgments are inferior to objective rulings in your game's reward system, you might want to explore a more cut-and-dried approach, such as that espoused by the Success Reward pattern.

Consequences

The Narrative Reward pattern is a potent means to encourage players to express themselves in offbeat and original ways. It promotes synergism between players that can help transform what would otherwise be a dry repetitive gaming session into an enthralling story. However, the interest of many gamers in role-playing lies in tactical competitiveness, rather than narrative brilliance. And, the inherently subjective nature of the rewards provided by this pattern may not sit well with players seeking clear-cut objective rules that "realistically" simulate real-world events.

Implementation Concerns

Psychological studies have shown that the longer the delay in providing a reward after observing a desired behavior, the less effective the reward. This is one reason why cigarette smoking is so addictive. The delay between taking a draw on a cigarette and the nicotine "hit" provided to the brain is a mere fraction of a second. So, the human brain closely associates smoking with pleasure even though the act of smoking is highly detrimental to the smoker.

What this tells us is that the shorter the delay between witnessing a desired behavior, such as the speaking of a witty or insightful comment, and its corresponding reward, the more potent will be the reward system. Since players must provide some verbal description of their character actions during conflicts and those are quickly followed by some form of resolution to those conflicts, it makes sense to apply the narrative reward to the conflict resolution in some fashion. Some games reward good narration in this way. That's not to say that narrative rewards must be applied to conflict resolution. Merely that this is a convenient and effective way to obtain the desired outcome. Alternately, the reward could come in the form of adding to some resource controlled by the player.

If a single die is rolled to resolve conflicts, a bonus could be applied as a reward to give the narrating player a greater chance of success in his endeavors. If the game uses a

dice pool system to resolve conflicts, the reward could take the form of adding a number of dice to that pool to similar effect.

Samples

Suppose we're designing an American Wild West style game named "Pig's Eye" where we want players to tell tall tales about their characters and use poker hands to resolve conflicts. In our game, at the start of each "Hand," each player is provided with 5 poker chips to add to his pile. The "Dealer" (see the Game Master pattern) then deals out five cards to every player, including himself, representing a poker hand. Play starts by having each player "ante" one chip and describing his character's basic action in the current conflict. The "ante" action may be interesting, but cannot break any fundamental principles of "reality." Play then proceeds around the table with each player placing "bets" from his poker chips based on the quality of his hand in a normal poker game fashion. Each time he "ups" the bet, a player can more fully describe what his character is attempting, with the following exceptions:

1) His actions must be spoken of in the past tense, as if the player was the one acting out the character's part.
2) His actions must be made more outlandish than any of those coming before in the same hand but is subjectively constrained by how much the player "upped" the current bet. These descriptions may ignore common sense or what most players would consider physically possible. The player must be careful concerning how outlandish his actions are, however, as the odds of "beating" the Dealer's hand decreases if he "stretches the truth" beyond what the Dealer considers acceptable given his current bet (more on how this happens later).
3) The description leaves the conflict's outcome unresolved.

Of course, like normal poker, players can participate in the game even if they have poor hands by bluffing. Players who do not want to bluff or risk further bets can "Fold," meaning that they can no longer modify their character's actions in the current "Hand."

Every time a player gives a particularly imaginative or exciting description of his character's actions, the dealer provides a narrative reward by dealing him one, two, or three cards based on how entertaining he feels the narration to be. These are incorporated into the player's hand. He must then discard an equal number of cards and give them back to the dealer who places them in a discard pile. The only player who does not get this possibility is the Dealer, who deals himself new cards only when a player narrates an action that is not sufficiently supported by the bet he has placed.

Once all bets are placed, the winning hand is determined by normal poker rules. The winner has the choice of either adding the pot to his poker chip pile or "cashing in" the chips by narrating the final outcome of the scene. If he narrates, he must take into account all previously narrated character actions and his description cannot result in the deaths of any player character. If he elects to collect the pot instead, the Dealer narrates the final result under the same restrictions.

Example of one player's narration for a hand:
Round 1: (Jesse antes 1 chip) "I aimed my pearl handled revolver at the gunslinger's hand." *Earns no cards from dealer for narration.*

Round 2: (Jesse starts out the round by betting 2 chips) "Well, I told you I was aiming my revolver at the gunslinger's hand, but I forgot to mention that I was doing this while leaping over the railing into the water below." *Dealer gives player 1 card for interesting narration.*

Round 3: (Jesse meets the current bet and ups it by 10 chips) "The water I told you about before was actually a shot glass full of whisky. After shooting and completing three summersaults in midair, I dove in head first. This was Mad Joe's bar, you know. Mad Joe was always known for his generosity in pouring drinks and I was counting on that fact to save my neck." *Laughing out loud, Dealer gives player 3 cards for narration but deals himself one card because he feels the three summersaults went a tad too far for upping the bet by 10 chips.*

All of his stellar narration won Jesse a full house and the hand. He decides to narrate rather than collect the pot: "Despite my mid-air acrobatics, my aim was so good that my bullet actually stuck in the end of the gunslinger's six-shooter, rendering it useless. And, Mad Joe's famous shot glass was so full that I never touched bottom. In fact, it took me a full minute just to swim back up to the surface."

Known Uses

InSpectres is a game that is completely up front about its "Ghost Busters" origins and theme. Players portray agents of an InSpectres franchise, a corporation specializing in the identification and elimination of any and all embarrassing supernatural infestations. The game incorporates a mechanic known as the "Confessional." At any time, a player can have his character "break the fourth wall" and speak directly to the players (not the other characters). At this time, he can provide additional information about current events, foreshadow future events that the players have not yet experienced, and even add traits to other characters. If a character is given a new trait in a Confessional and the character's player then portrays that character with that trait as described, the players are rewarded with an additional "franchise die" at the end of the current adventure. This is important, because franchise dice are the only way in which characters gain access to more resources. Individual characters do not increase in ability in any permanent way.

My Life with Master has participants portray the minions of Evil Masters. It uses opposed dice pools of d4s for conflict resolution (see the Opposed Rolls, Dice Pools, and Conflict Resolution patterns). All 4s are treated as 0s and the results of both sides are added and compared. During each scene, the game allows the Game Master to award a single player one more die that is a d4, a d6, or a d8 to add to their pool. This award is earned by narrating interesting accounts of character actions based on specific rules. The size of the die added depends on whether the detail includes Intimacy (d4), Desperation (d6), or Sincerity (d8). Since only a single die can be awarded on any given scene, a player demonstrating Desperation will win the die away from anyone already having earned a d4 for Intimacy. Intimacy involves actions such as the giving of a gift, sitting down together with another person, sharing a glass of wine or food, or physical contact, such as placing a hand on a friend's shoulder. Desperation is demonstrated by some emotional plea, "Master, please don't ask me to steal the cathedral's offering plates! Surely my soul would be damned forever. God could never forgive such an act!" Sincerity involves some great personal revelation exposing the character's deepest concerns. The game flatly states that the Master of the game is incapable of Sincerity, so the other characters can always trump the Master in this regard.

Sorcerer uses a Dice Pool mechanic during conflict resolutions. If players incorporate novel and interesting descriptions for their characters' actions, they can be awarded any number of additional dice to the roll. This "piling on the dice" can take a situation from which characters have almost no hope of winning and transform it into an event where they have a high probability of succeeding. These Narrative Rewards are given by the Game Master according to basic guidelines:

- A dramatic or appropriate quip while announcing the task: +1 die,
- The announced action moves the plot along significantly: +2 dice,
- Obstructive, petty announced action: -2 dice.

Success Reward

Intent

Provide players with incentives to find ways to have characters overcome obstacles.

Also Known As

E.P., X.P., EXP, Character Points, Development Points

Related Patterns

Attendance Reward, Failure Reward, Idiom, Narrative Reward

Motivation

The Success Reward pattern awards players for character success, whatever "success" means within a game's context. A game follows the Success Reward pattern if all of the following apply:

1) Character actions or goals may succeed or fail.
2) The chances of character success or failure can be influenced by player decisions.
3) A reward is given to the player if he succeeds in having his character achieve a goal or perform an action, but that same reward is withheld if he fails in that endeavor. (Note that a *different* reward may be given if the character fails without violating this pattern. In that case, the Failure Reward pattern is also being used.)

So, a Success Reward gives players a reason to want their characters to "win."

Applicability

You should consider the Success Reward pattern in your game if your goals include:

1) A desire for players to strive to have their characters "win" at times. (If a Failure Reward is also provided, that just means that a player must choose whether "winning" or "losing" is more desirable at any given time. This choice will be highly dependant on in-game circumstances and your overall game design.)
2) You are not averse to including a mechanical means of gauging the value of character success verses the value of character failure.

If you want players to have a real choice between characters winning and losing, you either need to forego the use of Success Rewards or balance them with appropriate Failure Rewards.

Consequences

A game incorporating Success Rewards equates character success with player success. Such a system will push players toward a competitive attitude where winning some final victory for their character is their primary goal. If this is an important design requirement, than Success Rewards alone may be exactly what you need. If not, this tendency can be counterbalanced with appropriate Failure Rewards. In other words, a game does not give players any real choice in their decision to have their characters win or lose with Success Rewards alone. It only gauges their cleverness in attaining success.

Implementation Concerns

A great deal of psychological research has shown that the sooner a reward follows an action, the more effective the reward will be in encouraging the rewarded activity. This tells us that a Success Reward should have as short a delay as possible between the attained success and its resulting reward. Of course, this goal must be weighed against other considerations, such as the mechanical overhead the reward imposes on game play.

Samples

Let's design a reward system for a game in which characters are all horse-jockeys. We want a significant portion of the game to focus on player competition, in that players will actually compete with one another to win horse races. Each race consists of having each player make10 rolls of a d12. On each roll, players will decide their actions, which will result in modifiers to either their current actions or future actions. At the beginning of the race, all horses start out dead-even. For example, the following modifiers may apply:

1) Decide whether to hang back behind the horse in front of him or push forward. Hanging back on a roll gives a character a cumulative +1 bonus on subsequent rolls. Pushing ahead gives a character a cumulative -1 bonus on subsequent rolls.
2) Pass another horse on the inside lane or outside lane of another horse. Passing on the inside provides a -1 penalty to the passing roll, but provides a +2 bonus to the next roll. Passing on the outside provides a +1 bonus to the passing roll, but no bonuses to subsequent rolls.

We're not going to go into details about how the actual die rolls are resolved, because that's not the point of this exercise. The point is that these modifiers ensure that player decisions impact whether a character succeeds or fails in winning the race.

Now, we want to introduce a Success Reward into our game. To keep the race moving along at a brisk pace, we decide that we don't want a Success Reward to be given on every roll. Instead, we elect to give a reward at the end of the race so that winning actually matters to the players. After careful consideration, we decide to give players "Prestige Points." Rather than have a "winner takes all" kind of system, we choose to give players a number of Prestige Points depending on how well they placed in the race.

So, the winner gets a number of Prestige Points equal to the number of jockeys in the race. The character that came in second gets one less point. The next place character gets one less, and so on. So that Prestige Points have an impact, we decide to allow players to spend them in future races. 1 Prestige Point = a +1 bonus on any roll. Essentially, Prestige Points represent experience in learning how to win races along with the chance to ride better horses as Prestige is gained.

Known Uses

Donjon awards experience points at the end of every encounter where an opponent is defeated. The levels of all conquered foes are summed and divided evenly among all participating characters in the form of experience points (see the Level pattern). Experience points are also awarded for non-combat situations as "goal awards" based on the level of the "donjon" the characters are exploring. Additional awards are given to players who do something that others find amusing or entertaining.

Dungeons & Dragons 3.5 awards experience points for defeating monsters and overcoming other obstacles. Every monster and trap is given a "Challenge Rating," which represents the difficulty of overcoming it. This value is compared to a character's level (see the Level pattern) via a table lookup to determine the resulting experience point value for the creature or trap. If multiple characters helped overcome an obstacle, its experience point value is divided accordingly. (The experience point values for characters of differing level must be calculated individually.)

Rolemaster Fantasy Roleplaying awards experience points for successfully performing actions. The total number of experience points determines a character's level (see the Level pattern). Awards are given for "maneuvers" (such as movement over difficult terrain), casting spells, taking and delivering critical wounds, killing, ideas, and travel. The basic amount of experience awarded is determined by various table look-ups on an action-by-action basis. This reward is modified by various factors, such as the condition of the foe when combat began and how "routine" the activity has become.

Warhammer Fantasy Role Play awards experience points for meeting "objectives." Major objectives earn characters between 100 and 200 points while minor objectives are worth 10 to 50 experience points. Major objectives include goals such as "foiling a sinister plot to overthrow a town's council" and "eradicating a force of raiding goblins." Minor objectives include "following a trail of clues to the next town" and "bribing a guard to gain entry into a warehouse." Experience points are spent by players to buy new "skills" (see the Gifts pattern), increase "scores" (see the Attribute pattern), and progress to new "professions" (see the Class Tree pattern).

Role-Playing Patterns

Faction

Intent

Segregate characters into opposing groups to promote in-game conflict.

Also Known As

Clan, Code

Related Patterns

Alignment, Idiom

Motivation

A faction is a group or category to which characters belong. A faction may be represented as an organization, such as a mob family or police department. Or, it may be abstract, as is represented by the concepts of "good" and "evil." What is important is that factions adopt codes of conduct, either explicitly or implicitly, that come into opposition with what the game's other factions consider acceptable. Then, when a player has a character perform acts in accordance with his faction's code, a counter response is demanded by characters of other factions. This generates conflict to drive the story forward.

In a game with many factions, it is possible for different factions to have common goals at times. In these cases, two or more factions may actually cooperate with one another for their mutual interests. However, a well-designed faction system ensures that some factions exist that will take exception to virtually any action. So, even though some factions may cooperate with one another on occasion, their collective opposing factions are likely to unite against them.

For example, in a modern-era game exploring the inhumanity of America's underworld, a terrorist organization might link up with organized crime in order to obtain black-market weapons. Neither of these groups trusts one another, and neither would go out of its way to support the other. They might even come into conflict with one another on occasion. But, the terrorists' need for weapons and the black-market's need for cash bring them together to serve their mutual goals. However, an arms deal of this nature is likely to attract the attention of not only the local police force and the FBI, but also that of the CIA and Homeland Security.

Applicability

Conflict is the heart and soul of role-playing games. The faction pattern does a good job of generating inter-group conflict but does a poor job of generating inter-personal conflict. If your game is more about person-to-person conflict rather than group-to-group conflict, you may want to avoid introducing factions into your game.

Consequences

The Faction pattern introduces tension between characters in different factions merely by the fact that the characters belong to different opposing groups. When characters perform actions that support their own factions, they often oppose the interests of other factions and conflict inevitably results.

Implementation Concerns

The Faction pattern does its job of promoting conflict between groups. Characters in different factions *will* conflict with one another. So, if you want to encourage *cooperation* between player characters, you need to either avoid using this pattern altogether or ensure that player characters all belong to the *same* faction. At the very least, you need to restrict characters to a sub-set of highly-similar factions that generally get along well, even though minor differences will inevitably cause some degree of conflict to arise, however small.

You also need to specify each faction's code of behavior, making sure that characters following that code necessarily oppose the interests of other factions. This demarcation is critical. Suppose you hear of a game where characters belong to law enforcement agencies such as the FBI, CIA, local police, or Homeland Security. Would that tell you enough to let you know where the game's major source of conflict arose? Hardly. Such a game could be designed such that all player characters are member of the "Law Enforcement" faction, working harmoniously in conjunction with one another to oppose crime in all its forms regardless of the specific organization to which each character belonged. Or, it might be all about inter-agency competition where gaining credit for a "collar" was all-important, regardless of which agency actually deserved it. In such a game, the actual solving of a crime might be of secondary importance. Most likely, though, it would fall somewhere in-between these two extremes.

Factions can have either a minor effect on your game or a major one. This ultimately depends on how much you reward players for having their characters support their factions in opposition to other factions. The rewards can come in many forms. However, if you want a character's faction to have a significant influence on how players role-play their characters, make sure that the rewards given for supporting a faction are either

1) Entirely independent of other reward systems.
2) Or, the primary means of obtaining a given reward.

Once again, if you want factions to have only a minor effect on play, you probably should reconsider using this pattern altogether. After all, why introduce complexity into your game for a minor benefit? Strip it out and focus the rules purely on the game's central core.

Samples

Suppose we want to create a game about the American Civil Rights movement of the 1960s. In doing so, we could create several factions modeled after historical groups involved in the movement and require that each character in our game be a member of one of them. Some of the groups and their beliefs might be described as follows:

Followers of Martin Luther King, Jr.

1) All races are equal.
2) All races should live together peacefully.
3) Violence is unacceptable as a means to victory.

Followers of Malcolm X

1) All races are equal.
2) The black race should form its own separate nation.
3) Violence is acceptable as a means to victory.

The Ku Klux Klan

1) The white race is superior to all other races.
2) Blacks and whites should be segregated.
3) Violence is acceptable as a means to victory.

Known Uses

Nobilis has five "Affiliations" that act as factions. These are "Heaven," "Hell," "Light," "Dark," and the "Wild." Each of these factions has its own "Code" and these Codes come into direct conflict with those of other factions. For example, the highest principles of each of these codes are

1) "Beauty is the highest principle." (Heaven)
2) "Corruption is the highest principle." (Hell)
3) "Humanity must live, and live forever." (Light)
4) "Humans should destroy themselves, individually." (Dark)
5) "Freedom is the highest principle." (Wild)

The primary reward system of the game involves awarding Miracle Points to players for having their characters follow their code in situations where doing so involves conflict.

The World of Darkness actually contains two forms of faction in **Vampire: the Requiem**. One faction involves a vampire's "Clan," which represents a character's particular breed of vampire. Players must choose one of five clans: "Daeva," "Gangrel," "Mekhet," "Nosferatu," and "Ventrue." (Clans also follow the Template

pattern, in that they provide characters with skills.) Each of the clans has "Stereotypes" of how they view the other clans. These stereotypes are universally derogatory, although each clan varies in the degree to which it looks down upon the others. The other faction category involves a character's "Covenant." A covenant is a vampire organization or government. Players may choose from "The Carthinians," "The Circle of the Crone," "The Invictus," "The Lancea Sanctum," and "The Ordo Dracul." Each of these covenants has its own aspirations and is involved in political maneuverings against the others. So, the game is brimming with built-in conflict.

Idiom

Intent

Provide players with a character aspect that mechanically rewards them for role-playing a previously declared personality type.

Also Known As

Humanity, Path, Spiritual Attributes

Related Patterns

Alignment

Motivation

The Idiom pattern rewards players for accurate portrayal of a character's persona. A game follows the Idiom pattern if it does the following:

1) Provides characters with a standard of behavior. (Whether that standard is chosen by the character's player, another player, the entire gaming group, or the game itself is irrelevant to identifying the pattern.)
2) Gives characters a gauge whose value raises or lowers based on how well the characters' actions conform to that standard. Commonly, values are numerical but this is not demanded by the pattern.

Idioms may be associated with attributes, traits, skills, or any other graduated gauge used in a game. Although the Idiom pattern does not require it, some games also restrict the situations to which an Idiom's value applies, such as when a character's morals or emotions come into play. Thus, players are encouraged to seek out situations in which the characteristic(s) apply.

The idea is to give players rational and potent reasons to portray their characters with real human emotions and personalities. Players following Idiom guidelines will continually seek out opportunities to act on their characters' emotions even if those actions would appear irrational to an outside observer. The Idiom pattern amply rewards players that pay attention to their character's passions and beliefs, so even near-

suicidal actions to attain selfless goals can make sense from a game mechanics standpoint.

Applicability

You should consider the Idiom pattern for your game if your goals include:

1) A desire to have players imbue their characters with strong beliefs and/or complex human emotions,
2) A willingness to give character "feelings" a real mechanical impact on game play.

Consequences

The Idiom pattern is a potent means to encourage players to portray character beliefs and emotions. In fact, players will often remain faithful to them even in circumstances where doing so has detrimental consequences to the character's well being. Such actions can have great dramatic impact, because they make a strong statement about what makes a character "tick."

Idioms can easily become a central focus of any game incorporating them. This quality can be quite useful if your game is about some core moral or emotional issue or strives to make each character's persona unique. However, its use could be detrimental to games having other agendas. As Jack Aidley observed in his game *Great Ork Gods*, "No Ork Is A Unique And Special Butterfly." In such a game, giving an ork an Idiom would merely distract play from the mayhem and destruction that the game is all about.

Implementation Concerns

To ensure that an Idiom has a fair representation of a player's role-playing history with his character, opportunities for Idiom values decreasing should approximately match the opportunities for it increasing. Incidentally, that does not mean that the values actually *will* decrease as often as they increase because player choice becomes the deciding factor on which direction it will go. What is important is that real choices be presented to players so that they have actual decisions to make concerning their character's actions.

If the Idiom is a resource, you should also strive to make role-playing opportunities for Idiom gain approximately equal from character to character. That is, a character with one type of Idiom should not automatically get opportunities to raise his Idiom by twice the value as characters with differing Idioms over an extended period. (If all characters have the *same* Idiom in your game, this point is moot.) Doing so will unfairly advantage the character with the rapidly advancing Idiom merely because of the Idiom he chose rather than from any role-playing merit. The character will be more successful because he will have greater resources from which to draw and the players of other characters may feel cheated. Unfortunately, you may find it difficult or impossible to balance the frequency of Idiom raising opportunities among all Idiom types. If this is the case, you can alleviate the problem by varying the number of points that can be

raised by a single Idiom raising opportunity or, if Idiom values are raised through some kind of random roll, by varying the odds of increasing a character's Idiom value.

Likewise, you should strive to make the opportunities to *apply* the Idiom's value approximately equal from player to player. Failure to do so will again unfairly favor one Idiom type over another, which could easily lead to many players selecting the same Idioms for their characters. If you want your game to support many different Idiom types, this can be a difficult issue to balance properly. Unfortunately, such issues are likely to remain hidden until some significant play testing of the Idiom system has been undertaken. Fixes for this problem include narrowing the applicability of "overly active" Idioms and expanding the applicability of others.

Samples

In a game about mythological heroes, the game could give each character an Idiom Attribute named Fate. One possible Idiom for such a game would be the following:

Tragic Hero

If you choose to follow this Idiom, you will portray your character with an eye toward his eventual, inevitable demise. Your character's primary virtue is courage and steadfastness in the face of great adversity. His greatest hope is to meet a fitting end that will do honor to his heritage. As such, he admires those who stand in battle and continue the struggle even after all hope appears lost.

Your tragic character will start his career with the inheritance of an ancestral weapon whose form you will choose to fit your character concept. If you decide it is magical or made of unusual or exceptional materials then it will always start out broken, being handed down from some great hero in your lineage. You can have the weapon re-forged as soon as you find someone capable of restoring it to its original glory and are yourself able to afford the required fee. Alternately, you and your Game Master may decide to have your ancestral weapon be lost or stolen, ripe for re-acquisition when your character attains sufficient power to seek it. The important restriction here is that your character does not possess a weapon with capabilities far beyond his ability to master or retain. The ancestral weapon is not an endowment meant to elevate your character's power far above the norm, but rather an interesting accessory intended to embellish your character's persona.

Actions Affecting your Character's Fate

A tragic hero will stand and fight no matter what the odds and will always revere his heritage. As such the following actions entitle a tragic hero to a roll to determine if his Fate increases by one point:

1) Battling courageously until his Hit Points fall to zero.
2) Doing some great honor to his ancestors, such as refusing to take permanent possession of a weapon of greater material worth than his ancestral one.

Similarly, the following actions will force a tragic hero to make a roll to determine if his Fate decreases by one point:

1) Retreating from battle no matter what the reason. (This includes "backing up" after having been directly assaulted so that the character can shoot his bow from the "back line." Tragic heroes handle their own melee affairs, thank you very much. Giving ground to draw an assailant along in order to gain some combat advantage is another thing entirely. Such actions incur no penalty.)
2) Dishonoring his ancestors in any way, such as failing to avenge any insult to his parentage or haggling too much over the price of re-forging his ancestral weapon.

When Fate Applies
If a tragic hero's Fate is positive, it applies to the following:

1) All Attack Rolls when wielding his ancestral weapon.
2) All Attack and Defense rolls in any battle where he continues fighting after his Hit Points have fallen below ½ maximum or where a comrade has fallen at his side due to injuries.

If his Fate is negative, it applies to all the following:

1) All Attack Rolls when wielding a weapon other than a family heirloom.
2) Rolls where he is not in a desperate circumstance. That is, when his Hit Points are above ½ maximum and no comrade has fallen at his side due to injuries. Even so, a negative Fate will never apply to a character's Attack rolls when wielding his ancestral weapon.

Known Uses

My Life with Master is a game where players portray deformed and unappreciated minions of Evil Masters. All characters have a "Love" attribute which players can raise by having their characters seek out and interact with their in-town "Connections" to demonstrate their inner humanity. The Idiom is crucial to the successful completion of the game, because accumulating enough Love is the only way a minion can disobey and overthrow his Master. The game ensures that every player has an equal number of opportunities to interact with Connections by keeping a fairly tight control over the sequence of scenes (see the Structured Story pattern), so all players have equal opportunities to raise their Love.

The Riddle of Steel has six "Spiritual Attributes" that follow the Idiom Trait pattern. (See the description of *The Riddle of Steel* in the Game Summaries section for why these are traits rather than attributes according to the definitions in this book.) These traits are "Conscience", "Destiny," "Drive," "Faith," "Luck," and "Passion." Each of these has specific rules describing when the trait ranks are raised and lowered and for when and how the trait values are applied. Players have some control over the

applicability of Drive, Faith, and Passion. Destiny might also fall under player control, depending on the flexibility of the "Seneschal" (Game Master). In any case, Drive applies when seeking some player defined "higher purpose." Faith deals with the character's religious beliefs, which are detailed by the player. Passion describes some great personal love or hatred toward a specific person or entity that the player elects. Passion applies when performing actions related to the beloved or despised subject. Players draw from all six Spiritual Attributes to improve their abilities (see the Resource pattern). This all-important characteristic makes Spiritual Attributes even more of a central focus of the game. The only way to raise Spiritual Attribute values is to demonstrate them through role-play.

Sorcerer characters are powerful humans that summon and bind demons to their will. The game gives each character a "Humanity" attribute that loosely follows the Idiom pattern. Exactly what Humanity represents is up to the gaming group, so players can explore various moral issues of their choosing. Humanity represents the very core of the game. "What are you willing to give up to get what you want?" Mechanically, Humanity is risked whenever demons are contacted, summoned, or bound and can be lost by inhumane acts (whatever the group decides constitutes inhumanity). If Humanity drops to zero, the player loses control of the character. The attribute can be raised by banishing sufficiently powerful demons and by acting humane (again, depending on the group's definition of what constitutes humanity). But, the analysis of Humanity is complicated by the fact that it is also conflicted. High values are good sometimes, but low values are good at others. So, it is somewhat inaccurate to describe a bonus to Humanity as a reward, making any classification of it as an Idiom problematic. Note that Humanity may also satisfy the Resource pattern, since it is gambled anytime a character contacts, summons, or binds a demon and gambling can be interpreted as a form of "spending." But, that is also debatable since it can be argued that performing any of these acts is inhumane and therefore these kinds of gambles fall squarely under the Idiom pattern already.

Story Patterns

Endgame

Intent

Provide rules that drive a story or character toward an end state.

Also Known As

The End, Happily Ever After, Finis

Related Patterns

Hit Points, Structured Story

Motivation

The Endgame pattern borrows its name from a chess term. In chess, the endgame is a phase characterized by the two opposing players having only their kings and one other major piece and/or a few pawns. This stage of the game quickly builds to the climax of checkmate or stalemate. Similarly, a role-playing game may incorporate rules to drive play toward some final climax. The climax can result in the end of a story, a character's participation in a story, or both. Note that, if the endgame results in a character's "retirement," it does not necessarily mean the character dies. He could simply ride off into the sunset, supposedly to live Happily Ever After.

The Endgame pattern is sometimes used in games having story and narrative as their primary focus. The ultimate aim of these games is to generate an interesting story with a dramatic climax. The point at which the endgame arises may be a purely player-driven decision based on whether he thinks his character is "ready" for it. Of course, this means that the player must have some viable means of making that judgment. Alternately, it may be triggered automatically when pre-specified events occur.

Applicability

The Endgame pattern is appropriate in games where you

1) Want to drive a story or character toward an end state.
2) Want characters to "survive" until that end state.
3) Want to ensure that end state results in a climactic scene or sequence.

Please note that the Trauma Gauge, Wound Trait, and Endgame patterns are not mutually exclusive. So, you might want to contemplate using some combination of these patterns in your game rather than rely on any one pattern in isolation. Of course, the Hit Points pattern provides an "end" for characters as well (death), so mixing it with the Endgame pattern might be counterproductive as it can interfere with the second goal listed above. If you're willing to accept that limitation or provide some means by which

death is not actually an "end" (such as a dead character continuing on as a ghost), an Endgame/Hit Points mixture is possible.

Since the Endgame pattern conveniently provides an ending to a story or character, any game designer using the pattern might reasonably want to "fill in" the other parts of the story as well. If that is your situation, you should consider using the Structured Story pattern as well. If your game goals do not include driving characters or plot to a definite end, you should avoid this pattern.

Consequences

The Endgame pattern provides a definite story or character end designed to satisfy player concerns.

Implementation Concerns

From a design standpoint, the pattern only requires that you let the players know when it is appropriate to enter into an endgame scenario, how the endgame is different from the rest of the game, and what that scenario will concern. If the endgame outcome is uncertain, it is a good idea to make sure that the players have a reasonable chance of success at this point, whatever that means based on your overall game concept.

Ensuring character survival until the endgame can be accomplished by purposefully omitting rules that kill characters or by explicitly incorporating rules that prevent character death until the endgame.

Samples

Let's create an example game using the Endgame pattern that could have easily (and less effectively) used some other means to measure character survivability. Suppose we are creating "Terror in the Skies," a game where player characters are all passengers on an airliner that is hijacked by a group of terrorists wielding machine guns, knives, bombs, and the like. Somehow, the hijackers got the whole works past security checkpoints. The terrorists threaten to kill one passenger every hour until their demands are met.

In our game, the endgame sequence will start as soon as the terrorists have murdered 20 passengers. Until that time, the player characters are completely immune to death. They can be shot, stabbed, gagged, threatened, and beaten, but they simply *will not die*. It is understood that the terrorists *will not* perform any actions that would necessarily result in their deaths, such as emptying a machine-gun clip into a player character's forehead. The terrorists are similarly immune from death and at least 3 terrorists must always be armed and wary until the endgame starts. Obviously, if the endgame is triggered when the 20th passenger is slain, non-player passengers and crew can die before the endgame commences.

When the endgame starts, the rules will state that all player characters and terrorists are at full capacity. If they were previously knocked unconscious, they will awaken. If they were shot, their wounds will not hinder them in any meaningful fashion (although

they will still exist as "dressing" and may cause the characters to limp or act in other appropriate ways). If they were bound, they will somehow escape their bonds. The objective of the scenes prior to the endgame is to build tension, demonstrate the evil intentions and conviction of the terrorists, and allow players time to assess the gestalt of the overall situation. They are not meant to hinder the characters. In fact, they may include scenes where some characters gain weapons of their own or one of the terrorists is convinced to betray his comrades. The endgame will be a rapid sequence of scenes in which the player characters attempt to retake the plane. During this phase, both they and the terrorists *can* be killed. The game ends completely when either all player characters die or the external door opens and the blood soaked heroes help each other out of the plane.

The characters survive until the endgame merely by the rules stating that it will be so. Of course, there are a lot of details to be worked out in exactly how characters can be killed after the endgame starts, but that is the purview of other patterns.

Known Uses

InSpectres is a game whose characters are ghost exterminators, much like the characters in the GhostBuster movies. At the beginning of every "mission", the game master decides how much the mission is worth in terms of "franchise dice". During the game, players earn these dice in conflicts. When all of the franchise dice have been earned, the GM declares the mission to be a success and rewards the players with the dice they have earned. These dice can then be spent to boost the resources of the ghost extermination business to help out in future "episodes".

My Life with Master has characters who are minions of Evil Masters. All minions have attributes of "Self-Loathing," "Weariness," and "Love." Masters have the single attribute of "Fear." The endgame starts when a minion successfully resists a command given him by his Master and his Love attribute is greater than the sum of his own Weariness and his Master's Fear. The endgame consists of a sequence of scenes culminating in the Master's death by the actions of one or more of his minions. After the Master dies, the fates of the minions are determined by other various formulas. Depending on circumstance, a minion may kill himself, integrate back into society, become an Evil Master himself, or meet some other end. The players narrate the details of how these results come about according to their own preferences.

Structured Story

Intent

Provide a means to ensure players always know what to do "right now." Also, generate a story plot following appropriate genre conventions building up to an exciting climax.

Also Known As

Structured Session

Related Patterns

Contest Tree, Endgame

Motivation

A structured story is a plot that is broken down into a number of different phases, each of which has a separate purpose within the game. The formulaic process keeps players from wasting time wondering what it is that they should be doing or descending into trivialities. This helps eliminate boring stretches where players aimlessly debate with one another about the course of action they should take: "I'm going to sell my loot" … "Sell your loot? What about the axe-murderer wandering about? We need to track him down *now*." "Well, yeah, but that would be easier if we had some cash…" Or, even worse: "So, what do you do?" "Ummm, I don't know, what do *you* think we should do…?"

The Structured Story pattern accomplishes its goal in the simplest way imaginable: it tells the players exactly what it is they are supposed to be doing at every point in the game. Games using the pattern don't tell players the specific actions they should have their characters perform, of course. Rather, they break stories up into sections that essentially say things like: "This is when you should equip your character for his upcoming tasks"; "This is when you should be out interviewing witnesses to the latest bank robbery"; "This is when the crew should plot their mutinee to seize the pirate's treasure"; etc. The pattern does not constrain the players in how they go about these tasks, though. It merely avoids discussions concerning what it is that the players should, in general, be doing.

The Structured Story pattern can be used to structure a game on a session-by-session basis. However, this is not a requirement of the pattern. It can just as effectively be used on stories that span multiple sessions.

The pattern can also be quite useful in emphasizing a game's genre conventions, especially those that re-play the same story concept time and again. Before you discard the pattern on the grounds that you don't want players limited in the kinds of stories they play out with your game, consider that most games do this anyway:

1. A villain does something evil.
2. The hero sets out to stop him.
3. The villain puts obstacles in the hero's path which the hero overcomes.
4. The hero kills the villain and gathers loot.
5. The hero goes back to town to await the next villain.

Sound familiar? The Structured Story pattern merely makes explicit what many games do anyway.

Applicability

Use the Structured Story pattern when

1) You want to avoid having players hem and haw about what their characters are going to do next.
2) You want your game to guide players through a pre-defined formulaic sequence of scenes.
3) The stories appropriate to your game's genre are easily broken down into identifiable phases.

If you want your game sessions to have a free flowing unstructured feel to them, you should not use this pattern.

Consequences

The Structured Story pattern accomplishes its primary goal of providing a simple framework to let players know what to do at any given time. However, it does this at the cost of imposing rigidity on the plot. Some players may find this inflexibility constraining and eventually tire of it.

Implementation Concerns

The Structured Story pattern has little subtlety to it. It lays bare the game's plot structure for all to see. So, you need to be very careful to make sure that the structure you impose strongly supports the genre of game you are designing rather than detract from it. Before deciding on what standard phases you are going to include in the plot structure, study the plots of several works (movies, short stories, and books) typical of the genre. Then boil the plot commonalities down to the bare essentials and use those as your story phases.

Samples

A game dealing with cops and robbers in a TV serial format might start each session with the discovery of a murder where the players are allowed to investigate the crime scene. After gathering clues and (possibly) discovering the identity of the victim, the session could then move to a phase where the detective characters interrogate various suspects. The questioning could lead to new clues and new crime scenes that, in turn, need investigation. Finally, when enough clues have been gathered to identify and/or locate the perpetrator, the game could proceed to a phase where the characters actually

apprehend the suspect. This might involve a car chase or combat scenario. The story might end there. Or, if the game had a "Law and Order" style format, play might then proceed to sentencing and trial phases. Once all of this transpired, if the game encouraged "sequels" or a series of stories, the next story would start once again with the discovery of a new murder victim. The point is that the sessions have a specific structure so that the stories the players create fit the dramatic tone and pacing of the game's genre and the players are never at a loss about what to do next.

Known Uses

InSpectres models its stories after the “GhostBusters” movies. Play starts with the characters being “interviewed” for their jobs within an InSpectres franchise. This gives the players the opportunity to “get to know” the characters a little before proceeding. Each story is thereafter set up with similar phases:

1) Getting “the call,” where clients hire the characters to solve some supernatural problem.
2) Research / Investigation, where the characters investigate the cause of the problem.
3) Suiting Up, where the characters procure the equipment they will need to deal with the situation.
4) Fieldwork, where the characters handle the problem.
5) Vacation, where the characters recover from their ordeal and await the next “call.”

My Life with Master models its stories after classic horror movies where evil masters command minions to perpetrate crimes against the local townsfolk for their own nefarious purposes. The players portray the minions and their goal is to eventually overthrow their master. The minions usually act alone, so scenes ordinarily involve a single minion and one or more non-player characters. Players act out scenes in a round-robin fashion. The Master will first order a minion to perform some evil deed, and the minion will go into town to carry out his orders. However, a player may request a scene with a “Connection” in town before he does Master’s bidding. A “Connection” is a non-player character to which the minion has a personal affinity and from which the minion acquires the “Love” he so desperately needs. So, scenes bounce from player to player with minions running around town either making overtures to connections or performing some evil act in servitude to Master. This flow of scenes is interrupted occasionally when triggered by various formulas provided in the game. One such scene is “The Horror Revealed,” where some horrible event transpires that is independent of the minions or the master. Another formula can trigger the “Endgame” (see the Endgame pattern), where a minion attempts to kill his Master.

Structural Patterns

Anonymous Rule

Intent

Hide game complexity and save paper or formatting space by incorporating rules of little importance within the text of more important rules.

Also Known As

Not applicable.

Related Patterns

Modularity

Motivation

An Anonymous Rule is a rule buried in the text of another rule or game entity. Such rules lack names of their own, and so are "anonymous." This does not describe the process whereby two or more rules are simplified through the creation of a single, more general rule. Rather, this speaks of the wholesale adoption of one rule into the description of another with little or no change to either.

Anonymous Rules often arise because RPG designers want two things:

1) Simple rules.
2) Rules that adequately cover the breadth of material pertinent to the game.

These goals are often at odds with one another. When these goals conflict, role-playing game design becomes difficult. To keep the number of rules to a minimum while simultaneously adding detail, some game writers add quirks and special characteristics to more significant rules. Frequently, these "quirks" could be partitioned as separate rules of their own but aren't because the game designer views them as too inconsequential to deserve their own heading. Headings take space. Space consumes paper. Paper costs money. We like money. In addition, merging a teensy unimportant rule into the context of a larger one seemingly simplifies the game, because players will then have fewer rules to learn.

Applicability

Using Anonymous Rules is generally a bad idea from a game design standpoint. In fact, it *almost* qualifies as an anti-pattern, or a pattern that should *never* be used. But, using an Anonymous Rule here and there is justified when you need to clarify some special situation in one text block that is not applicable elsewhere in your game. More extensive use can also be reasonable from a layout or marketing standpoint. If you are formatting your game and you encounter a situation where you absolutely *must* get your page count down and you can save a significant amount of real estate by merging one

rule into the text of another, it is logical to consider doing so. If you catch yourself doing this inadvertently, think carefully about your rationale before proceeding.

If you are trying to create a book with a strong emphasis on visual aesthetics, anonymous rules can help because they reduce the number of headings and other section breaks. Therefore, they enable text to flow more smoothly in a layout. Please note that we are talking about creating both a visual and literary work of art from the game prose, possibly elevating the work's appearance beyond that of a mere rulebook to that of a coffee table or art book. A stunning coffee table book left in plain sight can attract attention. Once it catches a person's eye, the artwork and poetic prose can lure him into browsing the book's pages, which will hopefully lead to hooking a player. If that is your tactic, then you can feel rightly justified in using anonymous rules despite the fact that doing so will obscure the game rules. Very few games have effectively pulled this off, though (Nobilis is a prime example of a game which has done so). You had better be a real pro in book layout to go down this path. Even then, weigh your decision carefully.

Consequences

Grafting one rule into another as an Anonymous Rule can indeed save a little space at times. Do not think that it can always save space, however. Sometimes, having multiple highly similar anonymous rules scattered throughout your game in various places can actually consume space. More importantly, anonymous rules *cannot* simplify your game. They might give the illusion that you have simplified things, but therein lay danger. Anonymous rules obscure game complexity, but they do not actually reduce it because all they do is hide rules, not eliminate them. The complexity is hidden not only from your players; it is also hidden from you as well. So, you are less likely to see the need for further simplification and will be less able to identify how you might go about it.

Anonymous Rules can also generate conflict between players when they cannot agree on what the rules actually state. "I'm sure I'm right. I saw that rule in here somewhere just last week. I just can't find it right now…"

Contemporary wisdom in game design states that a game should only contain features that focus on its primary purpose. A rule should always point the reader in the direction of what the game is actually about. If a rule is unworthy of a name, it is unworthy of your game.

Implementation Concerns

If you decide to incorporate an Anonymous Rule into the text of another rule, you should at least do something to highlight it so readers can quickly locate the information when necessary. Boldface or italics both work well for this purpose. If you graft multiple anonymous rules into the text of a single rule, number them or use bullets.

Samples

A game containing various fantasy races might describe a race of elves as follows:

Elf

Elves are thin and lithe faeries standing anywhere between 4 and 6 feet tall. They have pointed ears, fair complexions, and large almond shaped eyes that sparkle with mirth. Most of these generally social creatures live in wooded areas and excel in woodcraft. Their archery skills are renowned and some of history's most clever wizards and witches were elves. Nevertheless, elfin priests are often weaker than their human counterparts due to the race's flighty nature. The history of the long-lived elves offers much lore, artwork, and poetry to those deft enough to learn.

Elves can see well at night. On nights where the moon is out, elves can see with the same clarity as during daylight. On clear moonless nights, their eyes enable them to see up to a distance of 100 feet. Dark cloudy nights lower their viewing range to 30 feet.

In this account, the first paragraph is mere description. There is nothing in the text that affects the mechanics of the game. However, the second paragraph is an Anonymous Rule. The night vision rule could easily have been split out separately:

Elf

Elves are thin and lithe faeries standing anywhere between 4 and 6 feet tall. They have pointed ears, fair complexions, and large almond shaped eyes that sparkle with mirth and see keenly at night (as **Night Vision**). Most of these generally social creatures live in wooded areas and excel in woodcraft. Their archery skills are renowned and some of history's most clever wizards and witches were elves. Nevertheless, elfin priests are often weaker than their human counterparts due to the race's flighty nature. The history of the long-lived elves offers much lore, artwork, and poetry to those deft enough to learn.

Night Vision

Night Vision allows a creature or person to see well at night. On nights where the moon is out, beings with night vision can see with the same clarity as during daylight. On clear moonless nights, they can see up to a distance of 100 feet. Dark cloudy nights lower the viewing range of night sighted entities to 30 feet.

Not only does this partition the rules for Night Vision from the description of Elf, it also makes the gift conveniently available for other uses as well. If several races have this same characteristic, this separation can actually save space and, at the same time, make the game design more obvious to the reader. You might even say it simplified the game a smidge, since it took what was originally a rule exception ("Elves can see well in the dark and this is how it works") and transformed it into a standard gift ("Elves have the gift of Night Vision"). Assuming the game explicitly included the Gift pattern to begin with, adding one more gift to an already existing list does nothing to increase the overall game complexity. It merely adds detail.

Known Uses

Dungeons & Dragons v.3.5 has bulleted "Racial Traits" in its race descriptions (see the Race pattern). These are actually a mixture of anonymous and named gifts rather than what this book calls "traits" (see the Gift and Trait patterns). For example, the Gnome Racial Traits include "+2 racial bonus on saving throws against illusions" and "+4 dodge bonus to Armor Class against monsters of the giant type." Oddly enough, they actually name some of their Racial Traits, but then go ahead and embed the description for that trait directly in the text anyway: "Low Light Vision: A gnome can see twice as far as a human in starlight, moonlight, torchlight, and similar conditions…"

HARP has "Professional Abilities" (see the Gift and Skill patterns) incorporated into their "Professions" (see the Class pattern). The Professional Abilities section of every Profession is essentially a paragraph listing the effects of any number of anonymous gifts. For example, in the "Warrior Mage" description, it lists the Professional Abilities as: "Warrior Mages may learn spells from the Warrior Mage sphere. Beginning at first level, and then every seventh level thereafter (7^{th}, 14^{th}, etc.). Warrior Mages gain a +10 bonus to the weapon skill of their choice…"

Nobilis uses anonymous rules effectively to present the reader with smooth-flowing prose and a gorgeous layout. The work is undoubtedly one of the most attractive game books in existence. However, for all of its undisputed beauty, the book sometimes leaves the reader wondering what the game's rules actually are. For example, a Nobilis character's "Code" incorporates his belief system and is discussed at length. In the initial write-up of *RPG Design Patterns*, the fact that Nobilis rewards players for faithfully adhering to their Code (via Miracle Point awards) was omitted. This is a crucial role-playing reward that was overlooked until the oversight was corrected in a thread discussing morality and behavior mechanics on The Forge website. Even after learning of the mistake, searching for the detail via the book's index, and re-reading several chapters, the author still could not find the rule and had to ask for help in discovering it. To quote Tony Lower-Basch (author of Capes), "Page 133, right-most column, middle of the page. It's... oh... one sentence in the middle of a huge page of poetic description of Heaven and Hell. I'm not surprised you missed it." We quite agree. Most gamers would not take more than one or two gaming sessions to unearth all of the game's core concepts, especially if there were more than one player doing the reading. Beware, though. A game incorporating many anonymous rules that fell short of the Nobilis aesthetic would find many people losing interest before they fully understood the game.

Loose Coupling

Intent

Provide a means for one game concept to indirectly reference another game concept without tying one directly to the other.

Also Known As

Not Applicable

Related Patterns

Class, Gift, Rank, Skill

Motivation

This pattern gets its name directly from an important software engineering principle of the same name. Software designers often want various software modules to communicate with one another without either module depending on the implementation details of the other.

Similarly, game designers sometimes need to create relationships between various pre-defined game entities, such as classes, gifts, skills, and other abilities. In doing so, the game designer may seek a system for which future supplements can be dreamed up, written, and smoothly merged into the pre-existing game. For example, a fantasy game may be originally written with a single core rulebook containing some basic classes, skills, and spells. Depending on customer feedback, the writer may want to create supplements with more classes and spells but no skills. Or, he may find that creating a book of new classes is of far less importance to his customers than the creation of new skills and spells. The point is that, after the game designer gets the initial rule book out on the shelves, he wants the flexibility to expand the game in whatever fashion he deems necessary. And, he wants to be able to accomplish that task without having to go back and publish a second edition of the original book to accommodate the new materials. In handling these kinds of situations, the Loose Coupling pattern excels.

A game exhibits Loose Coupling if it sets up a relationship between two game entities whereby

1) Neither entity directly refers to the other.
2) Both refer to a third entity that acts as an intermediary.

One of the simplest ways to implement the Loose Coupling pattern is to have rules stating that one game entity has access to a certain number of a second game entity based on some criteria without ever mentioning specifics about the second entity. For example, a game using the Level pattern could state that a player can choose to give his character one skill for every level gained. This sets up an indirect relationship between

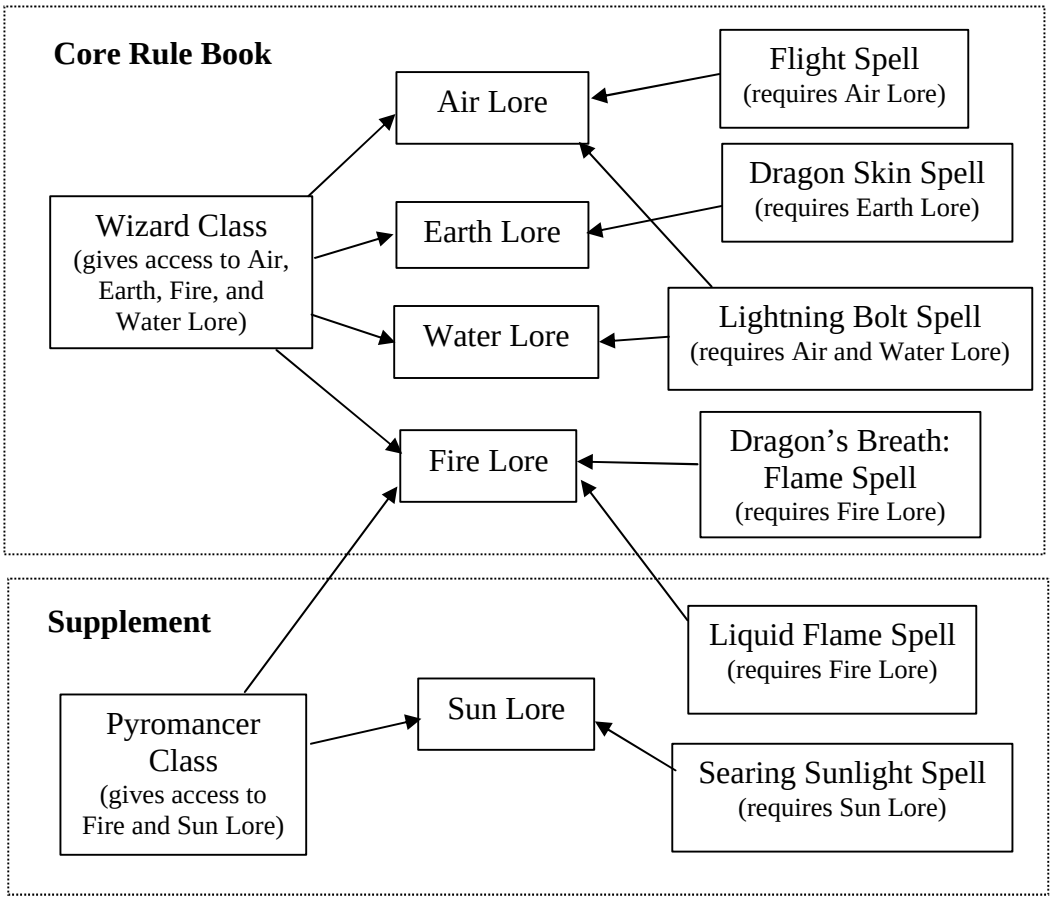
a character and his skills through the mechanism of his Level. However, game designers that want niche protection of character abilities may find this solution inadequate (see the Class pattern).

Niche protection can work in harmony with loose coupling by setting up indirect relationships through niche protected mechanisms, such as a rare skill, rather than common mechanisms, such as a character's Level. For example, various priest classes and a litany of religious spells could be related through an intermediate skill, such as Piety. If Piety is granted only to characters having a priest class, and all religious spells have a requirement that a character possess Piety to use them, then the religious spells have the niche protection the designer seeks. And, the designer can write up any new priest classes or religious spells he needs in the future without modifying what he has already written by virtue of the loose coupling.

If a finer granularity or a more expressive form of niche protection is desired, then the spells could demand more than one requirement from characters, which would be fulfilled in different ways by various classes. For example, if priestly battle spells were distinguished from priestly wrath spells, the spells themselves could specify additional requirements while still avoiding any reference to specific classes (which would destroy the loose coupling). Suppose the Battle spell Fortitude demanded a character possess 5^{th} rank in Piety as well as the gift of "Casting Battle Spells" to use. Suppose further that the class "Berserker" bestowed the skill of Piety and the gift of "Casting Battle Spells" but not the gift of "Casting Wrath Spells". Then, any character gaining the class of "Berserker" has access to the "Fortitude" spell (as soon as he attains 5^{th} rank in Piety). However, the Berserker character would not have access to wrath spells until he somehow obtains the gift "Casting Wrath Spells" elsewhere.

Example Structure

The following diagram illustrates how an initial Core Rule Book with a Wizard class and a number of spells can be extended by a supplement. The Wizard class of the original Core Rule Book allows access to various spells through the intermediary lore abilities, some of which are Fire Lore based spells. When the supplement is added, the new class of Pyromancer has access to the original Fire Lore based spells and the Wizard class has access to some new Fire Lore based spells as well. The new Sun Lore, though, gives the Pyromancer access to some new spells that the Wizard cannot learn.



Applicability

The Loose Coupling pattern makes sense when you want to

1) Design a game that can be easily extended with supplements.
2) Incorporate in the game design lists of pre-defined game components, such as classes, gifts, items, skills, etc.
3) Create flexible relationships between these pre-defined game entities.

If you are designing a self-contained single volume game for which you never intend to write supplements, the Loose Coupling pattern will be of little benefit. If game simplicity is a high priority, you may want to forego using the Loose Coupling pattern altogether. Loose Coupling introduces a layer of complexity that could be avoided by sacrificing flexibility and extensibility.

Consequences

The Loose Coupling pattern provides a powerful mechanism to create flexible relationships between lists of game entities. It does this at the cost of introducing intermediary components between the lists, and thereby increases game complexity.

Since the components at both ends of the loosely coupled relationship avoid direct references to the opposing end, players can become confused about what options are available to their characters. In the worst case, players are forced to scan through all available possibilities to determine their choices. This difficulty can be greatly mitigated through the use of other aids, such as tables and/or charts that explicitly list the options. As long as it is made clear that the tables do not represent all conceivable possibilities, merely those contained within a given text or supplement, the benefits of the Loose Coupling pattern remain unchanged.

Implementation Concerns

The Loose Coupling pattern introduces some complexity into a game in order to achieve its beneficial characteristics. However, it is impossible to use Loose Coupling to create relationships between all game entities, because loosely coupled components must, by necessity, directly reference some intermediary entity. So, in the interest of keeping your game complexity from ballooning out of control, it makes sense to decide up front what game elements need to be loosely coupled and which ones should use direct references.

You should pay particular attention to the entities that will serve as "intermediaries" in loosely coupled relationships. If you intend to create a single core rule book to which supplements will add material, you should do your best to include a sufficiently broad spectrum of intermediary objects in your core rule book to support the various supplements you foresee writing. If you do not have perfect foresight, though, you can still include those intermediaries in the supplements themselves. You risk annoying customers, though, if you create additional supplements based on those new intermediaries, because using the newest supplements demand both your core rule book and the supplements containing the intermediaries. Even so, it sometimes makes sense to purposefully segment entire game concepts into separate supplements, intermediaries and all. It all depends on how you choose to partition your game.

Samples

Suppose we want to design a game that incorporates psychic powers. We want to provide a wide range of psychic character types, each of which enjoys some niche protection from the others. So, we decide that we want to create a Loose Coupling between various psychic classes that we intend on writing and their powers. We decide to relate the classes and powers through various intermediary skills, which we are calling Psychic Disciplines. Here is an example of how a "Mentalist" class might be loosely coupled to a Psychic Power called "Mental Distraction" through the Psychic Discipline of "Mind over Mind."

Mentalist

A mentalist is a spiritualist who focuses on spirit, mind, and body. Although his efforts usually center on the discipline of his own soul and body, his powers extend to influence the minds of others as well.

Prerequisites
The character must attain 9th level in Mind over Body.

Psychic Disciplines: Mind over Mind (+4 rank), Mind over Body (+2 rank)

Mind over Mind

Cost per Rank: 16

Mind over Mind provides a spiritualist with great sensitivity to the thoughts and emotions of those around him. Beginners to this branch of spiritualism can do little more than sense the strong emotions of those around them. However a spiritualist who has gained great expertise in Mind over Mind can place thoughts directly into the minds of others and boldly walk past wary guards unchallenged.

Mental Distraction

Requirements: Rank 3 in Mind over Mind.
Affected Area: Up to one creature per rank in Mind over Mind.
Duration: 1 second.
Range: All targets must be within 10 feet per rank in Mind over Mind.

Mental Distraction creates a temporary distraction in the target's mind. The target experiences the distraction as a brief noise, such as a footstep or breaking glass. The noise can take any form desired by the psychic, but can never be made so loud as to cause the target discomfort. The origin of the noise is similarly controlled by the psychic, and is under no range constraints, since the noise actually exists only in the minds of his targets.

You should also note that this example not only provides a loose coupling between the Mentalist class and the Mental Distraction ability. It also provides a loose coupling between Mentalist and any other class granting the Psychic Discipline of Mind over Body (by virtue of the Prerequisites needed to obtain the Mentalist class).

Known Uses

Dungeons & Dragons v.3.5 uses Loose Coupling to specify the "Feats" that a character may possess (see the Gifts pattern). The number of Feats that a player may select for his character is dependent on a combination of his character's level (see the Level pattern) and his class (see the Class pattern). There is no niche protection of abilities granted through this relationship, as the character's level, an aspect common to all characters, acts as the relationship intermediary. However, some niche protection is provided by Prerequisites listed on the Feats themselves, such as a requirement of "12th-

level Caster." The loose coupling is somewhat tarnished by the fact that some of the character classes explicitly limit the selection of "Bonus Feats" (additional Feats granted by a class selection) to those on a given list. Aside from this discrepancy, the classes and feats do not otherwise directly reference one another and therefore remain true to the Loose Coupling pattern.

RIFTS relates character classes to skills (see the Class and Skills patterns) using the Loose Coupling pattern, although you have to really be looking for it to spot it. "Occupational Character Classes" (O.C.C.'s) bestow "O.C.C. Skills" that are directly listed in the class description, which is a form of "tight coupling." However, classes also have "O.C.C. Related Skills" and "Secondary Skills." These skills are selected by the player from various "Skill Categories" such as "Communications," "Mechanical," "Technical," and others. Although these categories frequently reference specific skills directly, they just as often appear with the descriptors of "Any" or "None." The descriptor of "None" means that the player cannot select any skills from this skill category if he chooses the class for his character. However, the descriptor of "Any" means that the player may choose any of the skills in the category, with the restriction that he is limited to a total number of skills over all categories.

This "Any" modifier is a well hidden example of Loose Coupling. In effect, it grants characters gaining the class an anonymous gift (see the Anonymous Rule and Gift patterns) allowing him to choose any skills in the specified skill category. By implication, all skills falling within a skill category require this anonymous gift as a prerequisite before a character can select them. The class and skill category are therefore related through this anonymous gift, which acts as the intermediary between the two. The loose coupling becomes obvious after noticing that the class does not restrict characters to a subset of the category, so the game author is free to create new skills in that category in subsequent supplements to which previously written classes automatically gain access. Similarly, the writer can create new classes in later supplements that access any skill categories he desires merely by allowing the class to grant the appropriate anonymous gifts.

Modularity

Intent

Boil each rule down to a single basic concern that it addresses. In other words, separate each concern into its own separate rule and give each its own unique name.

Also Known As

Don't Repeat Yourself (DRY)

Related Patterns

Anonymous Rule, Loose Coupling

Motivation

The Modularity design pattern splits different concepts out into separate rules. It is the antithesis of the Anonymous Rule design pattern. Its goal is to clarify the game's overall structure by giving each individual piece its own label. A game designer can thereby clearly see the components that make up his game. This may give the game designer insight into useful abstractions combining many disparate rules into fewer more general ones. In this way, a game can often be simplified and actually *reduce* its overall rule count. Modularity also frequently shortens the game text by allowing complex rules containing similar concepts to be broken down into simpler rules that merely reference common concerns. By splitting out a common issue into its own separate block, the concern can simply be referenced rather than have its text repeated in many places. This makes a game easier to modify and maintain, because an alteration to a concept requires a change in only one place. Finally, modular games are often easier to understand because the reader does not have to expend much effort in mentally identifying and separating important concepts into discrete elements. That work has already been done for him.

Applicability

The Modularity design pattern applies to any game where reducing overall complexity is an important concern. If layout and smooth text flow is a higher priority, you may want to consider the Anonymous Rule design pattern instead.

Consequences

Perfect modularity can result in an excessive number of individual text blocks unless careful attention is given to identifying useful abstractions combining similar issues into fewer more general rules.

Implementation Concerns

You can go a long way in modularizing your game by using the DRY principle of software development. Anytime you find yourself repeating something you've already written, split it out under its own heading and give it a name.

Samples

A game providing descriptions of various monster types might include the following description:

> ### Zombie
> A zombie is a member of the walking dead, a soulless, undead human that has risen from its grave to shamble aimlessly through the night. Its eyes always gaze downward with a glassy blank expression. This unconcerned stare is perhaps its most terrifying aspect, as the monster will rend and tear the flesh of anyone interfering with its nightly patrol without so much as an upward glance to indicate an awareness of its victim. Even as a zombie is disemboweled and dismembered, its stony countenance never wavers.
>
> *Like all undead, zombies are Immune to Mental Spells, are Susceptible to Holy Water, and have an Aversion to Sunlight.*

In this account, the first paragraph is mere description. There is nothing in the text that affects the mechanics of the game. However, the second paragraph (italicized) is a rule that would greatly benefit from being split out:

> ### Zombie
> A zombie is a member of the walking dead, a soulless, undead human that has risen from its grave to shamble aimlessly through the night. Its eyes always gaze downward with a glassy blank expression. This unconcerned stare is perhaps its most terrifying aspect, as the monster will rend and tear the flesh of anyone interfering with its nightly patrol without so much as an upward glance to indicate an awareness of its victim. Even as a zombie is disemboweled and dismembered, its stony countenance never wavers. Zombies have the standard strengths and weaknesses possessed by all undead (see **The Perks and Banes of Undeath**).
>
> ### The Perks and Banes of Undeath
> All members of the living dead are Immune to Mental Spells, are Susceptible to Holy Water, and have an Aversion to Sunlight.

Not only does this technique partition the resistances and vulnerabilities of undeath from the description of Zombie, it also makes the rule conveniently available for the description of other undead creatures. If we provide many such descriptions, this separation will quickly save space and, at the same time, make the game design more obvious to the reader. Finally, if we later decide to make all undead afraid of holy symbols, we can easily make this change to all undead by modifying the common rule.

Known Uses

Dungeons & Dragons v.3.5 has a section in the Dungeon Master's Guide called the "Condition Summary." This section is essentially a list of definitions for various forms of trauma that can affect a character. Among these are rules of "Blinded," "Deafened," "Dying," "Exhausted," "Sickened," "Stunned," etc. By separating these out, these various afflictions can be used throughout the game in a clear, succinct way.

Warhammer Fantasy Role Play has descriptions of many monsters in their Bestiary. In these descriptions, monsters have various forms of attack. Rather than endlessly repeat them for every monster description, the game has categories of various attack forms. Some of these are "claw," "kick/stomp," "gore," "constriction," etc. In the monster descriptions, these words are italicized to indicate that the words have special meaning. The definitions of the terms are split out into a separate section to which all monster descriptions refer. By doing this, the definitions can be clearly spelled out without needless repetition.

Priority Grid

Intent

Provide a means to design a game tool (such as a character) by forcing players to irrevocably prioritize various important concerns.

Also Known As

Not applicable

Related Patterns

Class, Gift, Resource, Skill, Trait

Motivation

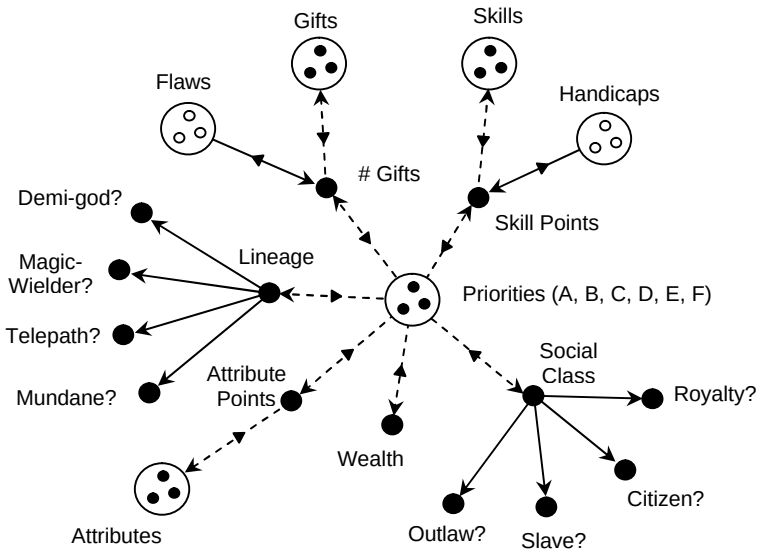
A Priority Grid allows a game designer to give players a simple means of designing game tools by trading-off features against one another based on their priorities. It is called a "grid" because the options are usually presented in a table format. Commonly, priorities are listed on the rows (such as A, B, C, etc.) and the features are listed in columns. (Obviously, the rows and columns may be interchanged without changing the pattern.) The player must select a limited number of A priorities, a limited number of B priorities, and so on until all priorities are expended. The primary advantage the Priority Grid has over other techniques (such as allotting players a resource to spend on features) is that a Priority Grid provides a means to force players to make extreme choices and it forces them to pick at least one choice at each Priority level. The Priority A choices may be far superior to the Priority B choices. The Priority B's may be far superior to the Priority C's, etc. For example, if a Priority Grid is used during character generation in a game where wealth and physical prowess are both valued (along with other concerns), such a system can make the choice of playing a crippled beggar equally as attractive as playing a warrior king, depending on the other trade-offs provided by the Priority Grid and the players' goals.

Applicability

Use the Priority Grid pattern when you:

1) Want to allow players to design their own game tools (such as characters, traits, classes, etc.) but force them to make extreme choices in doing so.
2) Have a fairly small number of features that you want players to trade-off.
3) The features to be traded-off can each be reasonably partitioned into a number of priority levels whose number does not exceed that of the number of features.
4) Each feature has an important game role so that player choices are not automatic.

Example Structure



Consequences

The Priority Grid Pattern provides a simple way to allow players to trade-off concerns in designing a game tool and allows the system to be set up in such a way that players must make extreme choices in one or more features. Supposedly, this will make the game tools more fun to use in game play. The pattern accomplishes this goal at the expense of some flexibility, however, since the degree to which a priority selection augments the game tool depends on a pre-defined table.

Implementation Concerns

By its nature, a Priority Grid makes it impossible for a player to select all of the Priority A choices in designing his game tool. This can be a problem if players believe that doing so is reasonable. Some text acknowledging this limitation and expressing the desire to have players make important choices can go a long way toward alleviating this concern.

If you decide to create a Priority Grid where the number of features exceeds the number of priorities, you will need to allow players to select multiple choices from one or more priority levels. For example, a Priority Grid with 5 Priority levels (I, II, III, IV, and V) and 8 features can work by giving players one each of Priorities I, II, III, and IV and four of Priority level V.

Creating a “balanced” Priority Grid, where no single option is the obvious choice in most cases, can be tricky. This is especially true if the grid contains some detrimental options that can be alleviated or overcome through play while others cannot. For example, suppose a Priority Grid was created to help design Super-Hero type

characters. One Priority A option may be a one-time chance to gain an extra "super-power" of some sort while another may be to start out with $1 Million. If super-powers cannot thereafter be obtained in play but wealth can be accumulated, then the super-power will likely be seen as more valuable than the financial incentive, no matter how large it is. Rather than a direct sum of money, suppose the grid provided a "Financial Savvy" rating that determined how quickly a character could accumulate money and that, once set, it never changed (or changed very little). Assuming money has sufficient importance in the game, an appropriately high "Financial Savvy" rating could easily be seen as being equal to a super-power.

Samples

In a game where players can design their own magical spell abilities, the following table might be an example of a Priority Grid where players select one each of Priorities A, B, and C and two of Priority D.

Priority	Effect(s)	Duration	Affected Area	Action Cost	Range
A	Select a single Category I Effect or two Category III Effects	1 hour per spell rank	40 foot radius or 1 creature per spell rank	1	100 feet
B	Select a single Category II Effect or two Category IV Effects.	1 minute per spell rank	20 foot radius or 1 creature per 2 spell ranks	3	50 feet
C	Select a single Category III Effect	10 seconds per rank	5 foot radius	7	20 feet
D	Select a single Category IV Effect	1 second per spell rank	1 creature	12	touch

Category I Effects: Puts target(s) to sleep, Blinds target(s), Levitates target(s) into air, Ignites target(s), Gives target(s) +5 roll bonus, Gives target(s) -5 roll penalty, Gives target(s) +5 armor bonus…

Category II Effects: Slows target(s) to ½ speed, Doubles target(s) speed, Gives target(s) +3 roll bonus, Gives target(s) -3 roll penalty, Gives target(s) +3 armor bonus…

Category III Effects: Trips target(s), Extinguishes all fires on target(s), Gives target(s) +2 roll bonus, Gives target(s) -2 roll penalty, Gives target(s) +2 armor bonus…

Category IV Effects: Warms target(s) against naturally occurring cold, Cools target(s) against naturally occurring heat, Gives target(s) +1 roll bonus, Gives target(s) -1 roll penalty, Gives target(s) +1 armor bonus…

Known Uses

The Riddle of Steel uses a Priority Grid in character generation. Players select one each of Priorities A through F in the categories of "Gifts & Flaws," "Proficiencies & Vagaries," "Social Class," "Skill Packets," "Attribute Points," and "Race & Sorcery." For example, if a player wants his character to be a "Landed Noble," then he must assign Priority A to his character's "Social Class."

Shadowrun also uses a Priority Grid during character creation. The player prioritizes "Race," "Magic," "Attributes," "Skills," and "Resources." The priorities determine whether the character can be a non-human (such as an elf) and how much the player gets to spend on skills, attributes, and cyberware.

The World of Darkness uses a very simple Priority Grid during character generation. It is simple enough that the rules do not actually provide tables. Nevertheless, it still follows the Priority Grid pattern. The game divides character attributes into "Mental," "Physical," and "Social" categories. Players must prioritize these categories and, depending on how they are ordered, gain a number of "dots" to spend on the attributes within that category. The same is then done for skills, which are divided up into the same three categories.

Game Summaries

Ars Magica (Fourth Edition)

Ars Magica was originally designed and written by Jonathon Tweet and Mark Rein•Hagen in 1987 and published by Lion Rampant. The fourth edition was published in 2003 by Atlas Games. *Ars Magica* is a game focused on wizards of the Order of Hermes in 14th century Europe (or, rather, *mythic* Europe). Each participant actually portrays a troupe of characters, including a Magus, one or more companions, and a number of grogs (henchmen). Since the game is primarily centered on magicians, most of the game's rules concern magic use. *Ars Magica* is considered by many to be the epitome of magic system design. At the very least, it sets a very high standard. Jonathon Tweet has stated on his website (http://jonathantweet.com) that the *Ars Magica* magic system forms the basis for the changes he made to 3rd edition D&D's magic system.

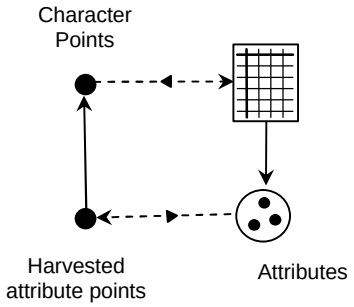
RPG Design Patterns Identified

Attribute ("Characteristic"), Experience Points, Faction ("Houses"), Flaw, Game Master ("Storyguide"), Gift, Hit Points ("Body"), Last Man Standing, Point Spend Gauge, Generalized Contest, Random Attribute, Rank, Resource, Skill, Template, Trait, Trauma Gauge ("Body" and "Fatigue")

Character Makeup

Ars Magica characters have eight primary attributes of "Intelligence," "Perception," "Strength," "Stamina," "Presence," "Communication," "Dexterity," and "Quickness." The values of these gauges usually fall in the range of -3 to +3.

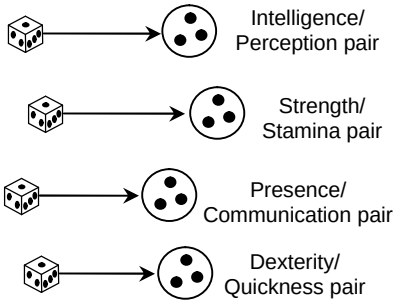
Setting Attributes (Option 1)



The game allows for two techniques for setting attribute values. The first is a Point Spend Attributes system where players are given 7 points to spend on attribute values. The costs of a given value depend on a table lookup. Players can earn more points to spend by accepting negative attribute values in some areas. The other technique is an interesting combination of the Random Attribute and Point Spend Gauge patterns. Here, pairs of different colored d10's are rolled. One d10 is designated positive, the other negative. The negative result is subtracted from the positive result. This gives the number of points that can be spent on pairs of attributes. This is done once for each of the following attribute pairs: Intelligence/Perception, Strength/Stamina, Presence/Communication, and Dexterity/Quickness.

Players select gifts ("Virtues") and flaws ("Flaws") for their characters. The only way to gain a gift is to accept flaws. Every gift and flaw has a point value. To be able to gain a gift, the same number of points must be earned by accepting its equivalent in

Setting Attributes (Option 2)

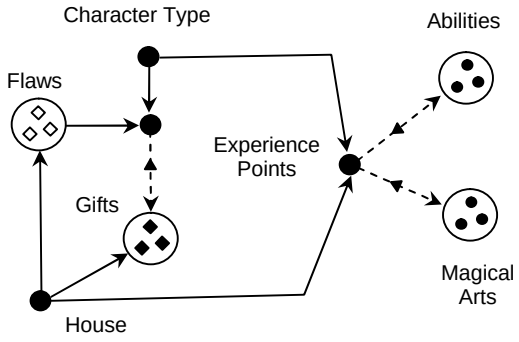


flaws. The total point cost of gifts and flaws a character can adopt depends on whether he is a Magus, a companion, or a grog.

Magi are assigned to "Houses." A House is essentially a sub-order within the "Order of Hermes" that specializes in various magical practices. When a player selects a House for his character, he adopts that House's template as well. Templates specify both required and optional gifts and flaws.

All characters have a list of skills ("Abilities") broken down into the categories of "Skills," "Talents," and "Knowledges." Skills are purchased with experience points. The number of experience points a character starts with is determined by his type (Magus, companion, or grog). A Magus character's House also has a big impact on the number of experience points with which he starts.

Gifts, Flaws, and Skills



Players purchase skill ranks with experience points as well. Rank costs are the same for all skills and are based on a table lookup.

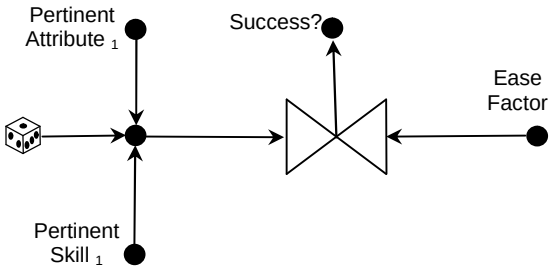
Magi have 15 additional attributes known as "Magical Arts," or just "Arts." These specify the mage's mastery of various arcane areas of knowledge. Five of these are "Techniques" and ten are "Forms." For flavor, these are (mostly) given Latin names. The Techniques are *Creo* (Create), *Intellego* (Perceive), *Muto* (Transform), *Perdo* (Destroy), and *Rego* (Control). The Forms are *Animal* (Animal), *Aquam* (Water), *Auram* (Air), *Corpus* (Body), *Herbam* (Plant), *Ignem* (Fire), *Imaginem* (Image), *Menem* (Mind), *Terram* (Earth), and *Vim* (Power).

What really makes Ars Magica's magic system so attractive is its flexibility. Rather than use magic skills directly, the game essentially introduces a skill *grammar*. The grammar is very simple. Every magical action, be it casting a spell or performing a ritual, must use one of the five Techniques as a verb and one of the ten Forms as a noun. So, Muto Terram (Transform Earth) could be used to change a boulder into a pile of sand. A character's rank in Muto Terram equals the sum of his ranks in the individual components of Muto and Terram. Pretty nifty. This generates a multitude of skill ranks with a minimum of bookkeeping. The core rulebook provides guidelines on what every combination of Technique and Form can accomplish.

Conflict System

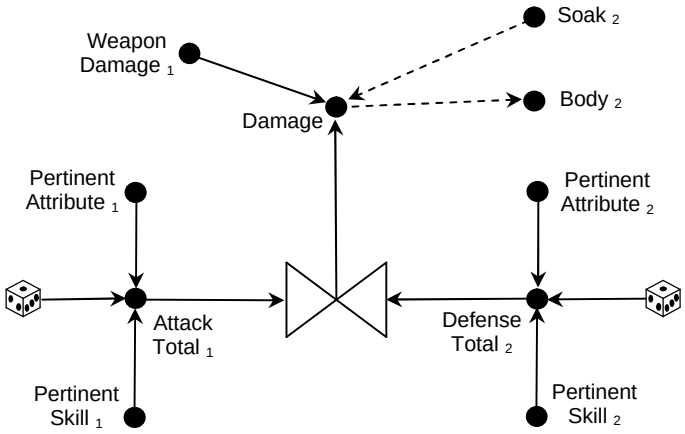
Ars Magica uses d10s on all contest rolls. Essentially all contests are resolved by rolling dice, adding pertinent attributes, skill ranks, and other modifiers to the rolled number, and comparing the result to a threshold, or "ease factor," which is assigned by the game master.

Conflict System (Using Skills)



There are three ways in which the actual dice are rolled, however. The simplest technique is used when the game master deems that there is no great chance for spectacular success or failure. In this case, a "simple die" consisting of a single d10 is rolled and the number is taken directly.

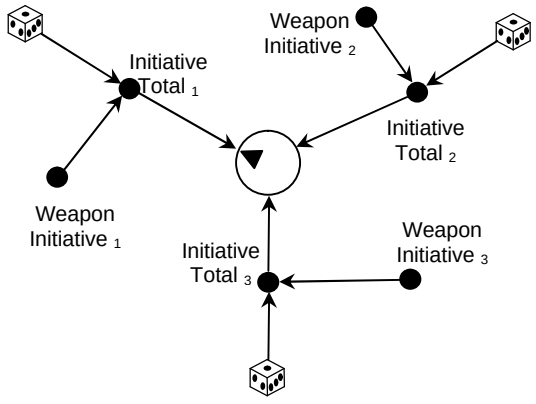
Conflict System (Combat)



The second case is adopted when a character has a chance of tremendous success, but no great chance for failure. In such cases, a "quality die" is rolled consisting of a single d10. If a 1 is rolled, an open-ended situation arises. The player re-rolls the die and doubles his result. As long as 1s keep getting rolled, the final multiplier keeps doubling.

The last case arises when a character is in a critical situation or is capable of both extreme success and devastating failure. This case, known as a "stress die," is identical to the "quality die" case except that an initial roll of 10 on the d10 counts as 0 and the character must roll again to determine if he "botches" the roll. If the die comes up as 10 again, the character has failed in a spectacular fashion. He must keep rolling "botch" dice as long as he keeps rolling 10s. Each additional 10 magnifies the

Turn Order (Movement Phase)

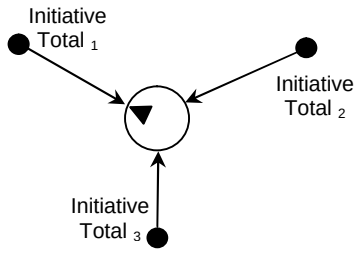


severity of the fumble.

In combat, both sides roll. The aggressor makes an Attack roll while the defender makes a Defense roll. If the Attack roll beats the Defense roll, the difference can be added either to the damage inflicted in the hit or to the combatant's roll on the next turn. The total damage equals the attacker's Damage rating (which takes into account the weapon and the character's strength and size). From this damage total is subtracted the defender's "Soak" rating (depending on his armor). For every five full points remaining, the defender suffers 1 point to his "Body" attribute.

Turn Order

Turn Order (1st & 2nd Missile Phase, Melee Phase)



Turn order in *Ars Magica* is complicated. Combat is broken up into rounds, each of which has six phases: "Movement," "First Missile," "Melee," "Second Missile," "Magic," and "Fatigue."

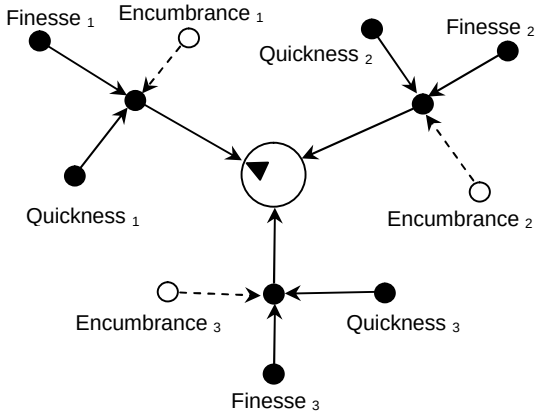
In the Movement phase, players make "stress rolls," adjusting their results by the Initiative score of the weapon the character is using. Characters move in order from highest initiative total to lowest.

In the First Missile phase, some range weapons fire. The range weapon type determines whether a character fires in the first, second, or both missile phases. The initiative totals rolled in the first phase are re-used to determine order of firing, if this becomes important.

During the Melee phase, everyone engaged in hand-to-hand combat resolves their attacks. Again, if it becomes important, characters with higher initiative scores perform their attacks before characters with lower initiative scores.

The Second Missile phase works just like the First Missile phase.

Turn Order (Magic Phase)



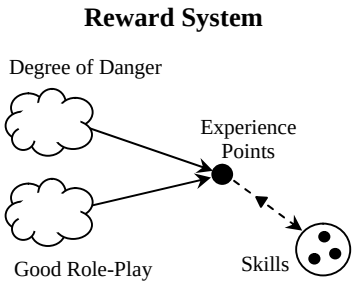
Note: Spontaneous spells go before formulaic spells, so the Magic Phase has two phases of its own.

Characters casting spells perform their actions in the Magic phase. All "spontaneous" spells go before "formulaic" spells. Otherwise, the order of actions is determined by a formula involving the character's Quickness, his rank in "Finesse," and the encumbrance of his equipment. Characters with higher values go before characters with lower values.

The final phase determines whether a character accumulates fatigue. Since this does not involve any character vs. character interactions, no explicit turn order is needed.

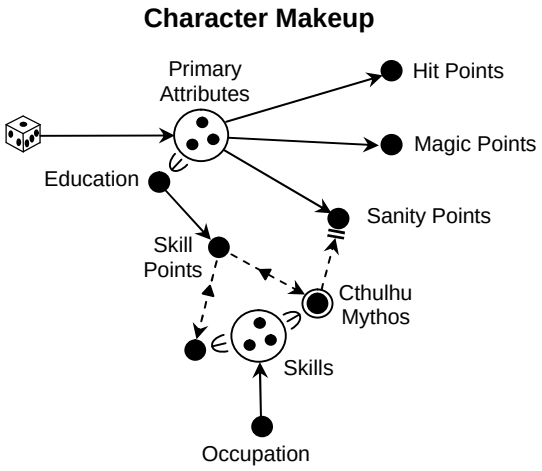
Reward System

Ars Magica rewards players through experience points. These are spent to raise skill ranks. The amount of experience points awarded by the game master is purely subjective. However, the rule book provides a number of guidelines on how many points are reasonable for different circumstances. These guidelines take into account the danger involved in a story and the quality of role-playing. In general, a player can probably expect to earn 1 to 4 experience points on every story.



Call of Cthulhu (Sixth Edition)

Call of Cthulhu is a classic game written by Sandy Peterson and Lynn Wilis and is published by Chaosium, Inc. It evokes the alien, macabre terror of H. P. Lovecraft, who is a true giant of the horror genre. Lovecraft's bizarre genius concocted a diverse mythology of ancient beings that supposedly traveled to Earth from far off galaxies where the very laws of physics and geometry differ radically from our own. The greatest of these colossi ruled the cosmos for billions of years, veritable gods in their potency and otherworldliness. The mere sight of such a creature can cost a man his sanity, if not his soul. Their longtime slumber is the only thing keeping these eldritch monsters from devouring the Earth, and various dark cults throughout the world seek their waking. Mankind might win small skirmishes to stave off the onslaught, but Earth's eventual pitiable fall is essentially inevitable.



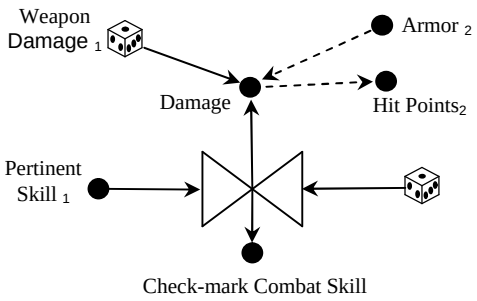
The game gets its name from one of Lovecraft's short stories, *The Call of Cthulhu*, which the rulebook includes to ensure anyone playing the game can taste what it aims to achieve. To reproduce the feeling of utter hopelessness in Lovecraft's tales, *Call of Cthulhu* describes most of the various aliens in terms that are well beyond the capabilities of the characters to handle. The weakest of them may be slain with great preparation and difficulty, but don't count on it. If you go into a Cthulhu adventure seeking to hunt down and kill the game's many-tentacled demons, your characters are going to live short, chilling lives.

RPG Design Patterns Identified

Faction (Cthulhu vs. everyone else), Game Master, Generalized Contest, Hit Points, Last Man Standing, Random Attribute, Rank (Skill percentages), Resource (Magic Points, Sanity Points), Skill

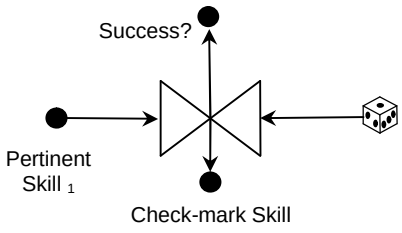
Character Makeup

Conflict System (Combat)



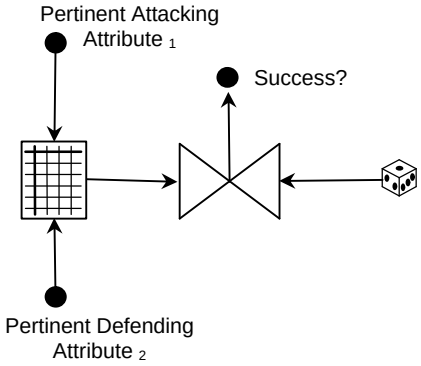
Characters ("Investigators") have 9 attributes whose values are generated randomly. These attributes are: "Strength" (STR), "Constitution" (CON), "Size" (SIZ), "Intelligence" (INT), "Power" (POW), "Dexterity" (DEX), "Appearance" (APP), "Education" (EDU), and "Sanity" (SAN). They also have three major resource pools from which to draw: Hit Points, Magic Points, and Sanity Points. These resources are determined by various formulae based on the attributes. Each character is assigned an "Occupation," such as "Antiquarian," "Dilettante," or "Professor." Each occupation is associated with a set of skills in which the player distributes a number of "Skill Points" whose quantity is determined by the character's Education attribute. The resulting values essentially set the percentage chance that a character has in succeeding at each skill.

Conflict System (Using Skills)



The "Cthulhu Mythos" skill is special. Gaining ranks this skill lowers the character's Sanity Point maximum. So, the more a character learns about the true and terrifying nature of the universe, the less stable he becomes. Sanity Points may vary anywhere from zero to the modified maximum. But if it ever drops to zero, the character goes completely insane. The mere sight of a Lovecraftian horror usually costs a character a portion of his sanity.

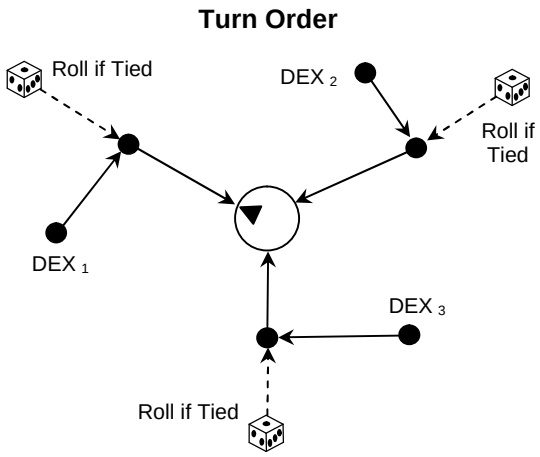
Conflict System (Using Attributes)



Conflict System

Each skill in *Call of Cthulhu* has a rank, which sets the percentage chance of the skill succeeding. To determine success, d100 is rolled. If the resulting value is less than or equal to the skill value, the skill succeeds, although a roll of 00 always fails. If the conflict involved weapons and combat, a successful roll indicates that damage is delivered to the target by rolling dice appropriate to the weapon used. If the target has armor, the damage is reduced by an amount depending on the armor's type.

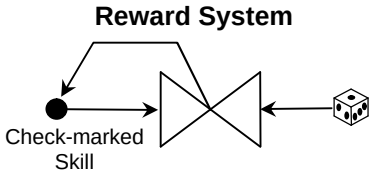
To resolve tasks involving attributes ("Characteristics"), players consult a table that pits one attribute against its opposing attribute. The table provides a percentage chance of the action succeeding.



Turn Order

Call of Cthulhu uses a karma-based initiative system in which character actions are ordered according to DEX, going from highest to lowest. In the case of a tie, the system becomes fortune-based with each player rolling d100 and going in order from lowest roll to highest.

Reward System



The first time a character successfully uses a skill in a dangerous situation on an adventure, he gets to place a check mark next to the skill on his sheet. At the end of an adventure, the player then determines if any of the check marked skills improve. To do this, the player rolls d100. If the result is greater than the skill's current rank, the player adds 1d10 to the rank of that skill. Thus, skills in which a character is a novice improve easily, but those in which he is expert rarely improve with use.

If the resulting percentage increase raises a skill above the 90% mark, the player gets to add 2d6 points to the character's Sanity Points. Joy of joys! Your character gets to stay sane a little longer!

Capes

Capes was written by Tony Lower-Basch and is published by Muse of Fire Studio. *Capes* is a super-heroes games with the premise: "Power is fun, but do you deserve it?" The game encourages players to explore their character's super-powers. However, whenever a super-power is used, the character incurs debt that will eventually degrade his performance. A character can lower this debt by performing acts that prove his worthiness to possess his powers.

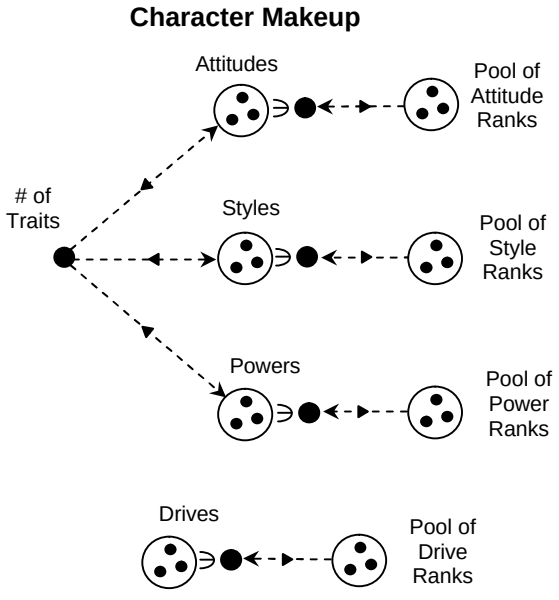
Heroes in *Capes* can literally *do* anything: lift buildings, fly between solar systems, see with X-ray vision, or anything else their hearts desire. That does not mean that a superhero can *accomplish* anything, though. The game is designed to allow players complete freedom in the description of their characters' actions. Its mechanics only come into play when opposing sides pit themselves against one another in accomplishing goals and seizing control of events.

RPG Design Patterns Identified

Conflicted Gauge ("Debt"), Contest Tree, GM-less, Idiom ("Drives"), Resource ("Story Tokens," "Debt," "Inspirations"), Negotiated Contest, Generalized Contest, Ranked Traits, Template

Character Makeup

Capes characters are made up of ranked traits. These traits are separated into four types: "Powers," "Skills," "Styles," and "Attitudes." Traits that are "Powers" are exclusively the domain of super-heroes. These include such things as super strength, the ability to fly, possessing the speed to dodge bullets, etc. "Skills" are abilities that ordinary people perform, such as "Painting landscapes" or "Programming computers." "Attitudes" describes a character's emotional tendencies, such as "Brooding," "Optimistic," or "I'll believe it when I see it." "Styles" pertain to how a character approaches a problem, such as "I always pull it out of the fire" or "Money is no object." Characters have twelve traits chosen by the player in the categories of "Attitudes," "Styles," and "Powers." The number of traits in each category must lie between 3 and 5. Once the traits are chosen, their ranks are assigned with sequential numbers (i.e., 1, 2, 3, …). The player decides which number is associated with each trait.



Important characters also have "Drives." These are separated out into "Heroic" Drives and "Villainous"

Drives. Needless to say, heroes have Heroic Drives while villains have Villainous Drives. The Heroic Drives are "Justice," "Truth," "Love," "Hope," and "Duty." The Villainous Drives are "Obsession," "Pride," "Power," "Despair," and "Fear." These are ranked, just like the traits. Some of these drives will be assigned an "Exemplar." An Exemplar is a character that embodies a Drive's concerns. For example, Lois Lane would be the Exemplar for Superman's Heroic Love Drive.

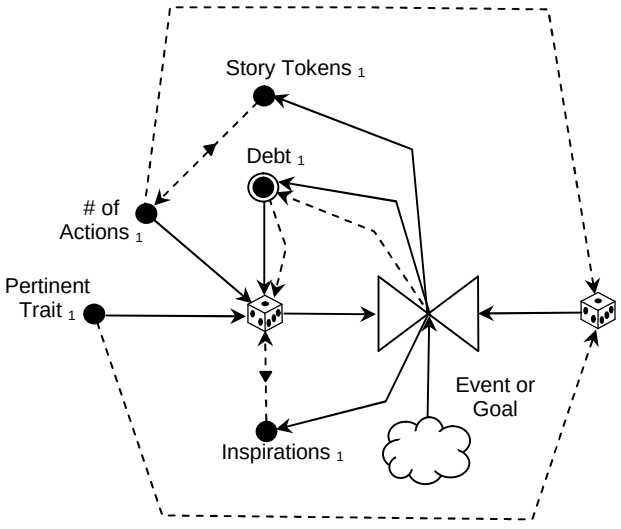
Conflict System

Conflicts are comprised of goals and events in *Capes*. A player may introduce a new event or goal in place of an action, if desired. A goal is some action a character wants to accomplish. The conflict resolution rules determine whether the action succeeds or fails. Events are incidents that are guaranteed to happen sometime in the near future. But, when first described, the event must leave open the resulting consequences. So, "the rope snaps" might be the declaration of an event. We *know* the rope is going to snap as soon as the event is declared, but we don't know what happens as a result. Players use the *Capes* conflict resolution rules to fight for the right to narrate what happens. This can be quite important to the story if, say, your character's lady love happens to be dangling over a boiling volcano by a rope. As a player, you will want to obtain the right to narrate the results of the event so that you can describe your character saving the day in the nick of time.

When a goal or event is declared, a short description of it is placed at the top of a 3x5 index card, which is then placed on the table. Two d6 of different colors are placed on the card with 1's showing. Each color represents one side of the conflict. So, each side has one starting die. Any player can declare an action for any of his characters in support of either side of a conflict. To do so, though, the character must use a trait having a rank greater than or equal to the number currently showing on the die he intends to roll. So, lower ranked abilities are used more at the beginning of conflicts while higher ranked ones are often saved for later. If a Skill is used (i.e., a non-super-power), it cannot be re-used in the same Scene. Super-powers *can* be re-used, but each usage saddles the character with a point of Debt.

Instead of rolling, a player can choose to spend an "Inspiration." Inspirations are resources that are each individually ranked from 1 to 6. Spending an Inspiration allows a player to set the

Conflict & Reward System



Note: The left/right mirror image also applies, but was omitted to simplify the diagram.

value of the die to the value of the Inspiration without rolling. So, they are highly desirable. Inspirations are gained at the end of conflict resolutions (as described below) to be saved for later use.

Unlike most games, a player can choose to roll any die in the conflict, whether it is one on his character's side or not. Players can either accept a roll or reject it. If rejected, the die is restored to its previous value. So, a player will generally accept higher rolls if he is rolling a die on his side of a conflict and he will generally reject higher rolls if he is rolling a die on his opponent's side.

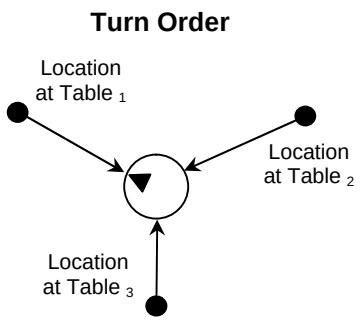
If a die roll is accepted, all other players have the opportunity to "react." A reaction is simply an action taken by a character to affect the specific die that was rolled. To be able to react, though, the character must use a trait having a rank greater than or equal to the current number on the die. If this is done, the reacting player gets to re-roll the die. Each player can only react once to a given roll.

A player may "Stake" some of his character's Debt on a conflict by placing Debt tokens on the 3x5 card representing the conflict. This is a gamble: the Debt doubles if he loses the conflict. On the other hand, if he wins, he rids himself of the Debt. The Debt tokens that were placed on the card are converted into Story Tokens, which are distributed to the player that introduced the conflict in the first place and to the winner of the conflict.

If Debt is gambled on a conflict, the gambler can "Split" the dice on the side of the conflict he is supporting. This gives that side more dice, which gives them a distinct advantage since the side with the largest total "controls" the conflict at any given time. Control earns narration rights for each action, but the conflict's final resolution can only happen under specific circumstances:

1) A player "claims" a side at the beginning of a round ("Page"). This is a declaration that he believes he will control the conflict at the end of the round.
2) The player controls the conflict (has the highest total on his dice) at the end of the round.

If these conditions are satisfied, the player wins the conflict and gains the right to narrate the outcome. As stated before, if the loser staked any Debt on the conflict, he gets double the amount back. The winner converts any Debt he staked into Story Tokens and gives them to the losers.



Turn Order

At the beginning of each scene, one player is designated the "Starter." This player sets up the scene before play progresses. Once the scene is described, players take turns going around the table in a clockwise fashion. At the end of every

scene, the person to the left of the current Starter becomes the new Starter and can set up whatever scene they like to push the story forward.

Reward System

Capes rewards come in the form of "Story Tokens" and "Inspirations." Story Tokens are earned by losing conflicts, so they are a Failure Reward. They can be spent to introduce characters into scenes (the first is always free) and to buy extra actions during conflicts.

As stated above, "Inspirations" are a resource that can be spent to set dice to specific values rather than gamble on rolls. Inspirations are each individually ranked. After a conflict is resolved, the winner matches the dice used on both sides to generate "Inspirations." The winner determines their ranks by pairing up the dice used in the conflict. The rank of an Inspiration equals the value of one side's die minus the value of the opposing side's die. The side with the larger die gets the Inspiration. So, both winners and losers can earn Inspirations from a conflict, depending on how the winner decides to match the dice up. Any dice that are left over because the opposing side has no die to match it are paired against zero. That is, the Inspirations gained from unpaired dice have ratings equal to the numbers on the dice.

Debt can also be considered a reward, since staking it on conflicts allows a character to split dice and more easily win those conflicts. However, if a character accumulates too much Debt, it starts to hinder his activities. So, Debt is actually a Conflicted Gauge.

Code of Unaris

Code of Unaris was written by Gary Pratt and is published by GoldLeaf Games. It is a game designed specifically to support role-playing on Internet chat sites. The game's premise is that a renegade computer program has inadvertently created a wormhole to an ancient past set in the world of Unaris. One billion years ago, Earth's moon was lush and fertile, supporting an ancient human civilization of medieval level technology and fantastic magic. The mysterious Internet program not only allows players to see into this remarkable past, but also allows them to alter it in small ways so that history may be altered. This is fortunate, because one powerful mage, known as The Warlock, has taken notice of the wormhole's influence. His goal is to eradicate mankind forever, so his discovery that mankind has survived a billion years is disconcerting to him. His knowledge of the wormhole, though, may allow him to correct his mistakes and make humankind's downfall a reality. It is the players' collective responsibility to see to it that this does not happen.

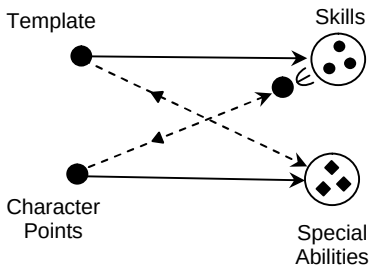
RPG Design Patterns Identified

Gift (Special Abilities), Hit Points (Life Points), Last Man Standing, Negotiated Contest, Rank, Resource (Hack Points), Skill, Template

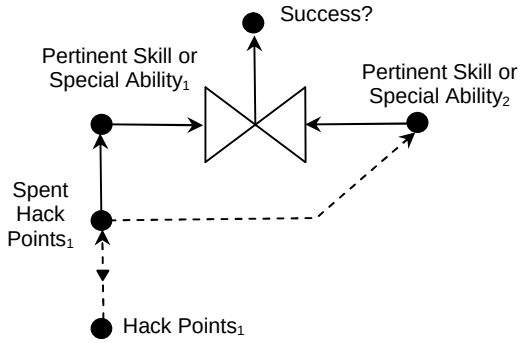
Character Makeup

Characters are created through the expenditure of "Bonus Points," or "Character Points." Generally, players start their character designs from given templates. Players thereafter spend their Bonus Points in customizing their characters. These points are spent primarily in gaining ranks in various skills, such as "Jump" and "Alchemy" and may gain "Special Abilities" (Gifts) such as "Cold Resistant" or "Quickdraw." Characters also have "Reputations," such as "Charming," "Hot-Headed," or "Snooty." Reputations provide modifiers to various pertinent skills. All characters start with a single reputation, but more can be accumulated (and/or dropped) as play progresses. Characters also have a number of "Life Points," which act as hit points.

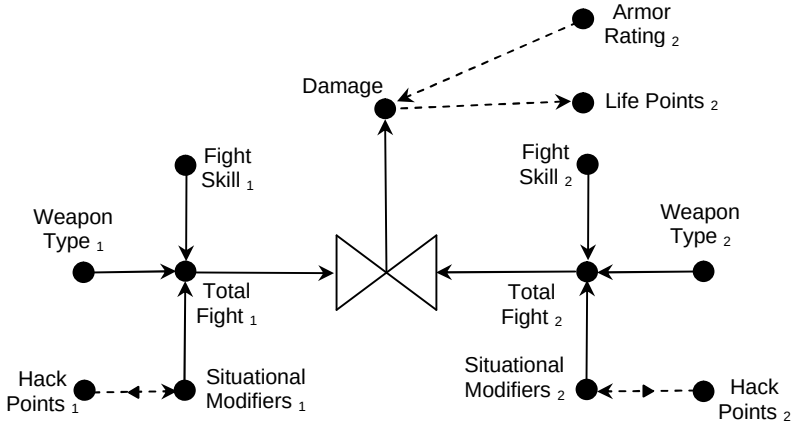
Character Makeup



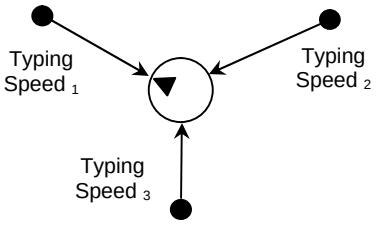
Conflict System (Using Skills)



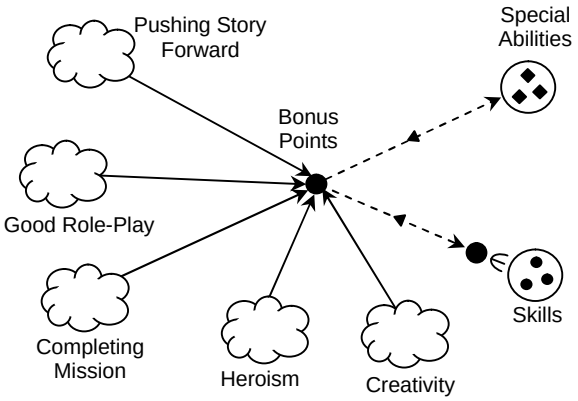
Conflict System (Combat)



Turn Order



Reward System



Conflict System

The most original mechanic in the game is known as "hacking." Hacking is a process by which statements of recent past events or circumstances stated by the Game Master may be altered. Every session, each player starts with 20 Hack Points. The alteration of a single word costs a player one Hack Point. Changing a quantity up or down by 50% costs a player 2 Hack Points. In this way, a player can alter the circumstances of the game world so that an unfavorable situation can be transformed into a favorable one. “A lit bomb lands at your feet” could potentially be hacked into “A lit torch lands at your feet.” Thereafter, conflict resolution is very simple. When one skill is pitted against that of an opponent, the higher skill rank wins.

Turn Order

Code of Unaris takes advantage of the fact that it is a game specifically designed to support role-playing in Internet chat rooms. The order in which actions are taken is the order in which they are entered into the chat session. Thus, the faster you type, the higher the priority of your actions.

Reward System

The game master rewards players at the end of a session by giving them Bonus Points. The amount of points awarded depends on if they completed the mission, how heroic the players portrayed their characters, how inventive they were in overcoming their opposition, for pushing the story forward, and for how well the character’s role was played.

Dogs in the Vineyard

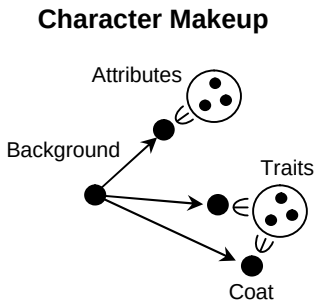
Dogs in the Vineyard was written by Vincent Baker. It is set in the mid-19th-century American West, in an area similar to pre-statehood Utah, dominated by a religious order known as "The Faith." The characters play pioneers who have been charged with the task of traveling from town to town to make sure their denizens stay true to The Faith's teachings. Any citizen falling into sin opens opportunities for the always present threat of demonic influence and possession. Left unchecked, the acts of a single sinful individual will lead others into an ever widening spiral of adultery, false doctrine, and murder which ultimately proves devastating to any affected community. Because of the dire consequences of failure, the characters are free to use any means to ensure the purity of the faith, including violence and execution. Their all-important charge earns the characters the title of "God's Watchdogs," or "Dogs" for short.

RPG Design Patterns Identified

Attribute, Conflicted Gauge (Fallout), Contest Tree, Dice Pool, Faction ("The Faith" vs. Demons), Negotiated Contest, Generalized Contest, Recycled Fortune, Structured Story, Trait

Character Makeup

Characters have 4 attributes ("Stats") of "Acuity", "Body", "Heart", and "Will", which are not rated as simple numbers but rather in terms of dice (d6s). So, a character might have Acuity of 4d6, Body of 2d6, Heart of 5d6, and Will of 6d6. Characters also have traits, which are similarly rated in terms of dice, although d4s, d8s, and d10s are allowed in addition to d6s on traits. Traits come in three flavors: action-oriented ("Traits"), relationship-oriented ("Relationships"), and item oriented ("Belongings"). To determine how many dice are distributed among attributes and traits, the player selects a "Background" for his character from a short list, such as "Well Rounded" or "Complicated History." One of a character's most important belongings is his Coat, which is adorned with unique colorful patterns stitched lovingly by relatives and friends. Dog coats are generally long, duster-style overcoats giving the wearers an Old-West gunslingers look.

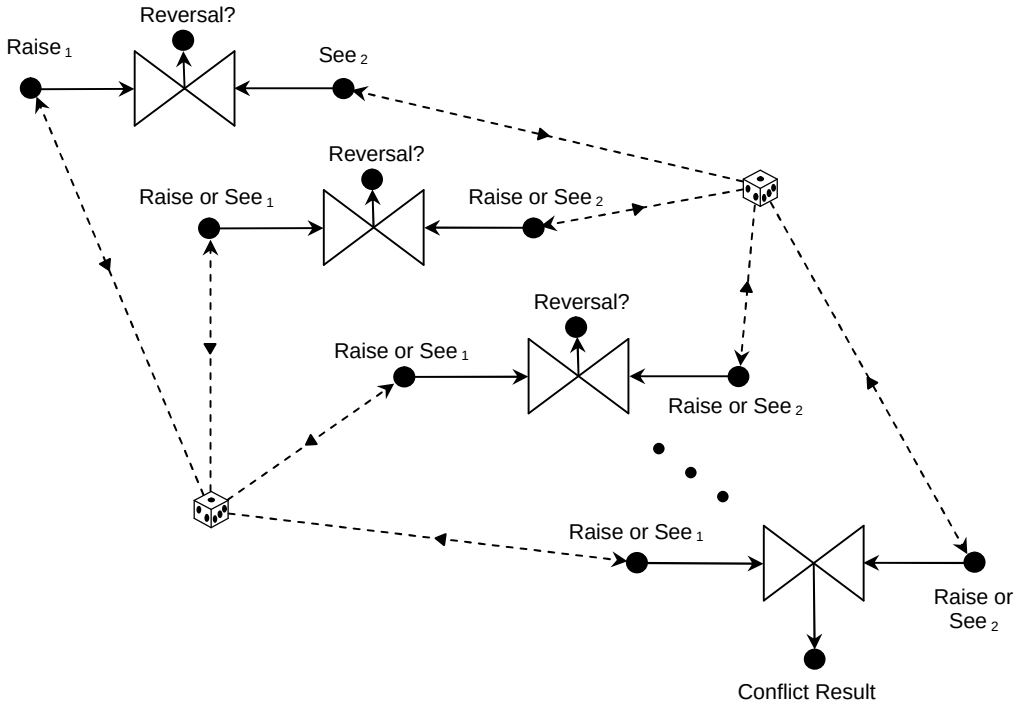


Conflict System

Conflict resolution uses dice pools. Unlike most dice-pool based games, the dice in Dogs pools are of varying sizes made up of d4s, d6s, d8s, and d10s. When a conflict starts in Dogs, the players must first determine exactly what is at stake in the conflict. The number and type of dice that are rolled on each side is determined by the actions being undertaken, which indicates what attributes and traits are brought into play according to the game rules. For example, if a character is just talking, he uses Acuity + Heart. If he is shooting a gun, he uses Acuity + Will, etc. Dice for traits are added

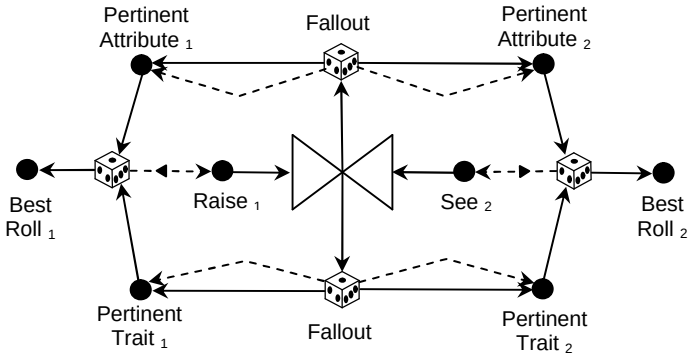
when they are brought into play. So, if a character attacks with his knife having a rating of 1d6, he adds 1d6 to his dice pool.

High-Level Conflict System



After the appropriate dice pools for all parties are gathered, everyone rolls their dice simultaneously and leaves them on the table so that everyone can view them. The person initiating the conflict goes first. Both sides then go through a bidding process similar to that of poker. The initiator goes first with a "Raise," putting forth two dice and describing the action those dice represent. The sum of the dice equals the value of the "Raise." The opponent then "Sees" with any number of dice of his own, stating how his character responds to the initial action. The sum of the "See" dice must equal or exceed the current "Raise" value. If this can be done with a single die, the opponent has achieved a "Reversal." If this is done with two dice, the opponent "Dodged" the attack with no harm to either side. If the "See" required three or more dice, the attack successfully lands, inflicting "Fallout Dice" to the defender.

Low-Level Conflict & Reward System



Note: Fallout was included twice to simplify the diagram.

The greater the number of dice needed to "See" the Raise, the greater the number of "Fallout Dice" sustained. The size of the Fallout Dice is determined by the type of attack. Verbal attacks deliver d4 damage while gunshots inflict d10 damage, etc.

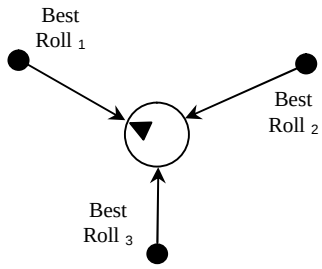
If a player finds himself losing the conflict, he can escalate it. Thus, a verbal confrontation between two friends may be escalated by having one or the other pull a knife. At this point, the person pulling the knife adds the dice associated with the knife to his dice pool as well as any pertinent attributes, giving him more ammo in the conflict. His opponent may respond by pulling his gun, adding the pertinent dice to his own pool as well. At any time, a player can "Give," meaning he concedes the conflict to avoid taking further Fallout Dice.

One basic rule of escalating conflicts is that the dice associated with a given attribute or trait can be added only once to a given conflict. So, while a knife or gun could be used in more than one action within the conflict, its dice are added only once to the dice pool.

Turn Order

When it isn't clear who is opening a conflict or when there are more than two parties to a conflict, the order is taken from highest to lowest "Best Roll." The Best Roll is simply the sum of the two highest dice in each pool.

Turn Order



Reward System

Oddly enough, the Fallout Dice inflicted on a character can act as a reward. When you roll Fallout Dice after a conflict, if any rolls of 1 come up, something good happens. The player can choose to raise an attribute by a point, create a new trait of 1d6, alter the die size of an existing trait, etc. Thus, Fallout Dice are conflicted. Whatever effects are chosen must be justified as a result of the conflict.

Donjon

Donjon was written by Clinton R. Nixon and is published by Anvilwerks. The name is a play on the word "dungeon" and the game is geared toward capturing the wonder and excitement of many-a-gamers' first experiences playing *Dungeons & Dragons*. Like that ground-breaking game, the primary purpose of a *Donjon* character is to beat the holy living heck out of any encountered monsters and loot them for booty. (Well, okay, maybe they don't have to kill *every* living thing they encounter, but the game definitely encourages bloodshed and mayhem.) The game has a brilliant twist that sets it distinctly apart from its predecessors. Players have as much input into the game world as the game master. If a player searches for a secret door and succeeds in his roll, he finds a secret door *whether the game master originally put one there or not*! If he listens carefully for noises and succeeds in his roll, he hears a noise *and can describe what it is that he hears*!

The player can introduce a number of facts into the world depending on how well he succeeds on his contest rolls. So, the better the "hear noise" roll, the more detail he is allowed to give to the noise he hears. In this way, he could potentially "hear" a troop of sleeping goblins. The game master "fills in" the surrounding environment with facts of his own as constrained by those stated by the players. So, while the character would indeed hear the sounds of multiple goblin snores, there is still a lot left unspecified in the encounter. If the player stated that there were many goblins and that they were sleeping, it's unlikely that he would be able to state what kind of treasure or weapons they possessed. That leaves plenty of room for GM input to challenge the characters. "Oh, too bad you didn't check for traps. A bell jingles when you open the door. Let's see if any of the goblins wake up…"

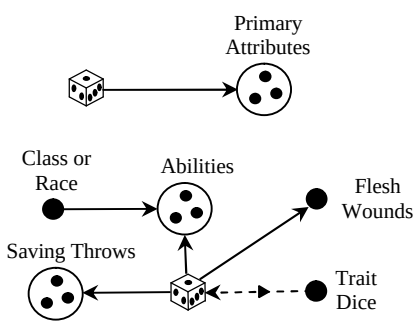
RPG Design Patterns Identified

Dice Pools, Experience Points, Gambled Resource Attribute (Wealth), Game Master, Hit Points ("Flesh Wounds"), Last Man Standing, Level, Negotiated Contest, Point-Spend Attributes (optional), Generalized Contest, Race, Random Attribute (optional), Rank, Resource (attributes act as pools that can be attacked), Structured Story, Template, Trait

Character Makeup

Characters have six attributes of "Virility," "Cerebrality," "Discernment," "Wherewithal," and "Sociality" which range in value from one to six. There are a number of ways to set these values, but the "Standard" technique is to roll them randomly taking the median roll of three d6. Characters also have either a "Class" or "Race" (never both). *Donjon* "Classes" and "Races" are actually templates, because they provide no niche protection, aside from that

Character Makeup

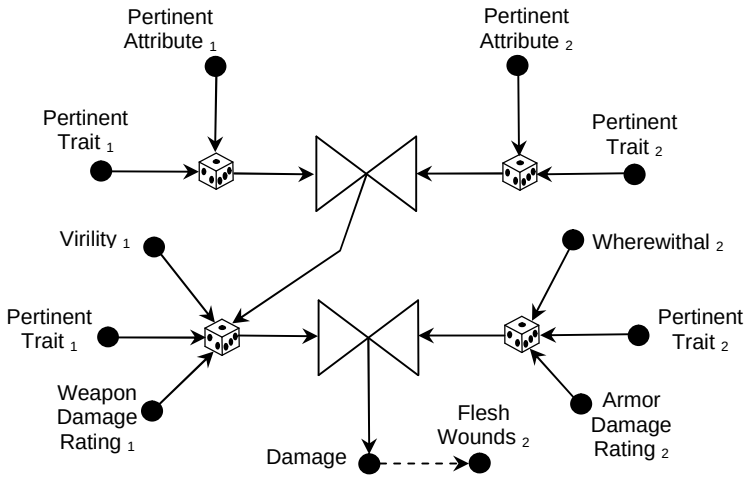


enforced by the game master. Rather, the rules specify how the players create their own templates by assigning traits ("Abilities"). There really is little difference between a "Class" and a "Race," but any character choosing a "Class" is assumed to be human and all members of a "Race" have the same traits. Characters are also given a "Flesh Wounds" attribute, which acts as hit points. Finally, character prowess is gauged in the forms of levels and trait ranks.

Conflict System

Donjon uses dice pools of d20s to resolve conflicts (other die sizes may be substituted if desired). The player rolls a number of d20s equal to his character's pertinent attribute plus his rank in any single appropriate trait. The game master does likewise and the individual numbers are compared. The side with the highest number on any die wins. The number of dice on the winning side having a value greater than the opponent's highest die indicates the number of successes (with ties counting as successes). Successes can be transferred to subsequent rolls (such as damage for an attack roll) in the form of additional dice. Once a damaging attack lands, a second conflict roll determines how much damage it delivers. The aggressor rolls a number of dice equal to the attack successes + Virility + Weapon Damage Rating + any pertinent trait rank. The defender rolls a number of dice equal to his Wherewithal + Armor Damage Rating + any pertinent trait rank. The successes from this second conflict roll are subtracted from the target's hit points ("Flesh Wounds").

Conflict System (Combat)

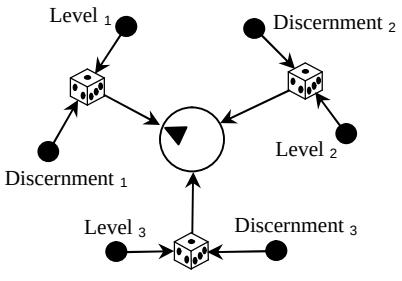


However, the most interesting use of successes resulting from a conflict roll is the ability to translate them into facts as described above. So, a success translates to either an additional die on a follow-on roll *or* a fact, whichever the player wants.

Turn Order

Donjon combat is divided into "Flurries." A flurry is a sequence of rapid attacks and counterattacks taken by both sides. To determine who goes when during a flurry, each player rolls a number of d20s equal to his character's "Discernment" attribute plus his Level. Each of these represents the timing of an action. The game master counts down from 20. Every time someone has an initiative die with

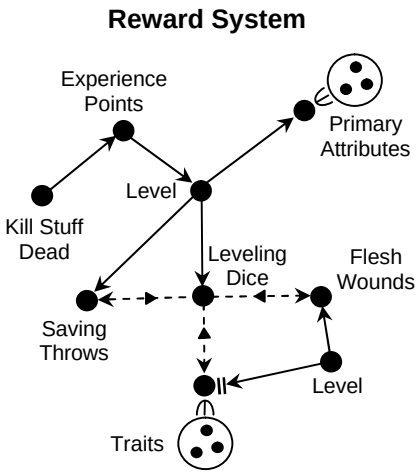
Turn Order



the stated number, his character is allowed to perform an action.

Reward System

Donjon rewards players with experience points when they kill things. The game is quite clear that no experience is awarded unless the creature is left in a bloody mound. If it somehow limps, crawls, or staggers away, as the Soup Nazi would say, *"No XP for you!"* The total number of experience points earned by a character determines his level. Every level he gains, a character gains 5 "Leveling Dice." These can be applied to trait ranks, hit points, or "Saving Throws," but the character's level determines the maximum number that can be spent in any area. In addition, characters gain bonuses to their attribute scores as they gain levels.



Dungeons & Dragons v.3.5

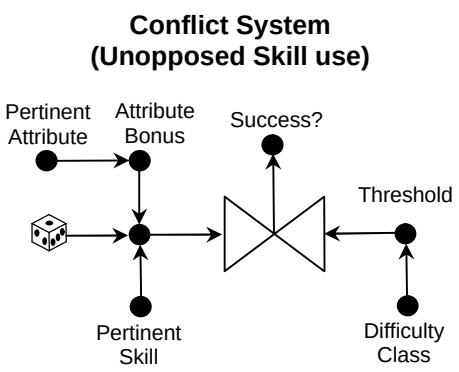
Dungeons & Dragons v.3.5, published by Wizards of the Coast, is the most recent edition of the venerated, much loved, and oft disparaged Dungeons and Dragons game (originally written by Dave Arneson and Gary Gygax). It was the first and is, without a doubt, the widest known role-playing game in existence. The game has evolved over the years, but the fantasy setting and game focus of the current edition remains faithful to its roots. The game is about pitting heroic characters against fantastic and fabulous monsters, beating the beasts to a bloody pulp, gathering their loot, and earning experience points. As play continues, characters accumulate treasure and power, thus allowing them to challenge tougher and better equipped opponents.

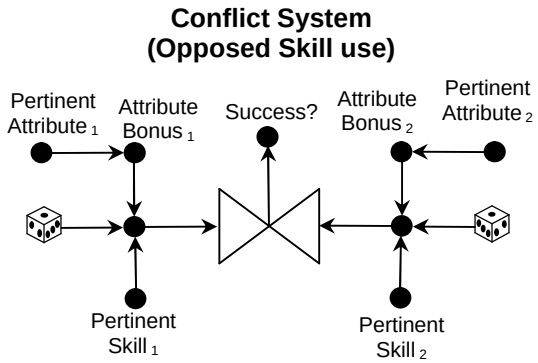
RPG Design Patterns Identified

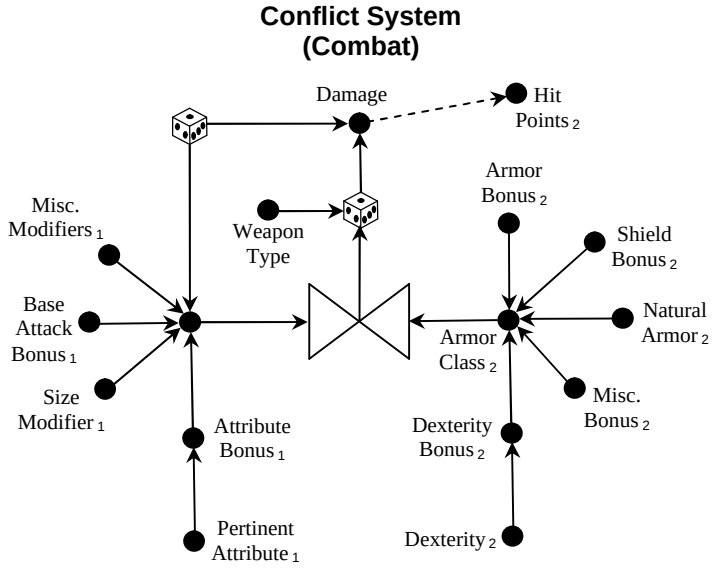
Alignment, Class Tree, Game Master, Generalized Contest, Gift, Faction (based on various alignment categories), Hit Points, Last Man Standing, Race, Rank, Resource (money), Random Attributes, Skill, Success Reward

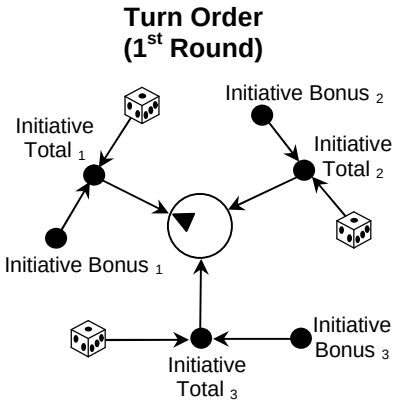
Character Makeup

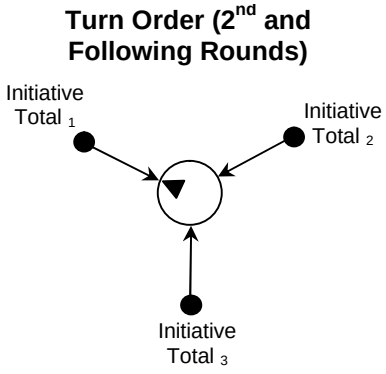
Characters have six primary attributes ("Abilities") of "Strength," "Dexterity," "Constitution," "Intelligence," "Wisdom," and "Charisma" whose values are generated by random dice rolls. Players must choose a race for their character as "Dwarf," "Elf," "Gnome," "Half-Orc," "Half-Elf," "Halfling," or "Human" (other possibilities also exist). The race provides adjustments to the basic attributes along with a variety of other gifts. Each character has a small number of classes, usually one, chosen from options such as "Fighter," "Monk," "Rogue," and "Wizard." Each class gives a character access to special abilities, which improve as the character gains levels by accumulating experience points. Skill ranks, whose costs are set by the character's class, and gifts ("Feats") are also gained as characters gain levels. Characters accumulate "Hit Points" as they gain levels by rolling dice and adding in adjustments for Constitution. The character's class determines the size of dice used.





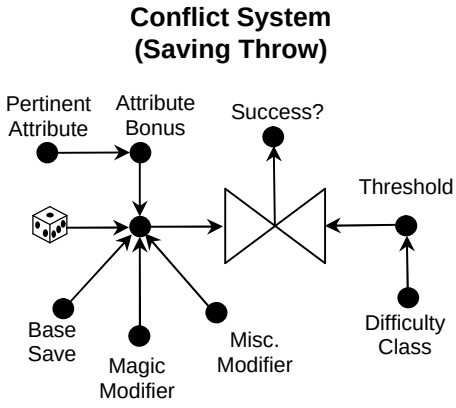






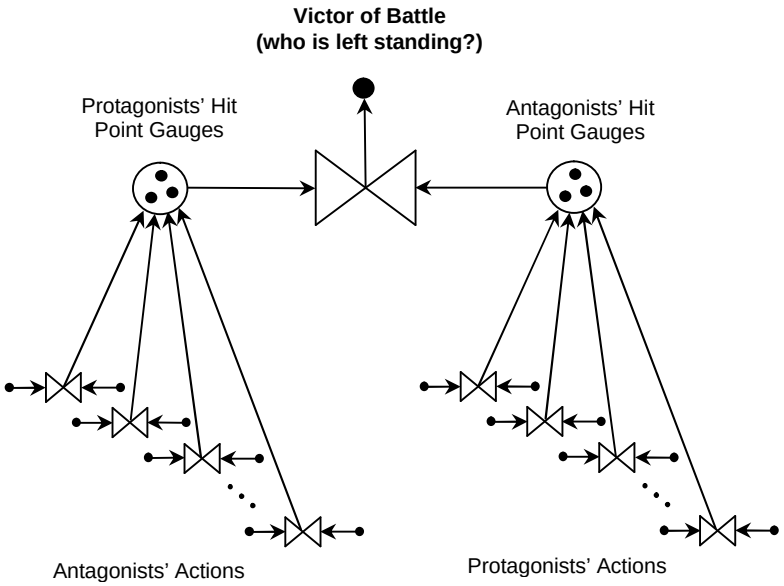
Conflict System

Task resolution is determined by having a player roll a single d20. When a character uses his skills, the player adds various attribute and skill bonuses, and compares the result to a threshold. The result indicates either success or failure.



If the character is attempting to strike an opponent with a weapon, various modifiers are added to his roll, including the character's "Base Attack Bonus," pertinent attribute bonuses, size modifiers, and other miscellaneous modifiers. This is compared to the target's "Armor Class," which comprises the character's "Armor Bonus,"

"Shield Bonus," "Dex Bonus," "Natural Armor," and other miscellaneous bonuses. If the blow lands, the player then rolls dice to determine damage depending on the weapon used. If the d20 roll was high enough, the weapon may do double or triple damage.

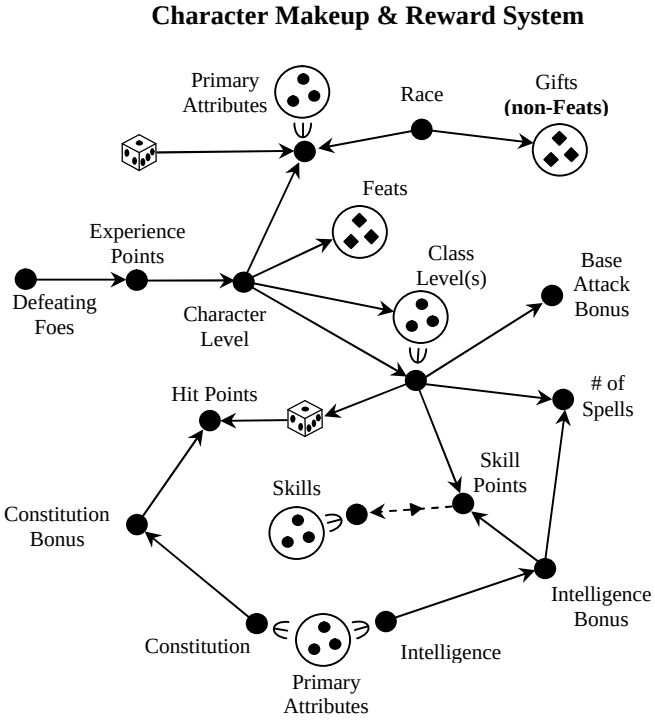


Turn Order

Initiative is determined by having each combatant roll a d20 at the beginning of melee and adding adjustments based on attributes and gifts. The order then proceeds in a round-robin fashion until the conflict ends. A host of rules control what actions (and how many) are allowed on each player's turn.

Reward System

D&D rewards players by giving their characters experience points and treasure for defeating foes. The experience points accumulate to determine a character's level. The treasure is either used directly, if it has its own in-game purpose (such as a Ring of Invisibility or a Wand of Fireballs), or is used to purchase items that enhance the group's ability to slaughter ever more powerful unsuspecting beasties.



Elfs

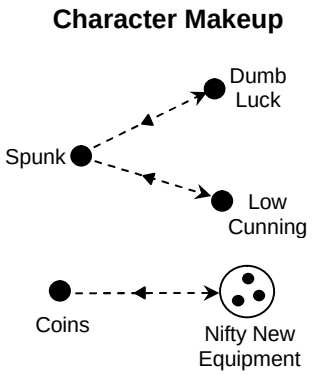
Elfs was written by Ron Edwards and is published by Adept Press. It is a light-hearted lark of a game where all characters are obnoxiously selfish pointy-eared elves, er, elfs. The game is a satire of traditional fantasy role-playing games that mercilessly pokes fun at the loot gathering behaviors exhibited by players of those games. Its most innovative feature is the way in which it separates character and player goals in conflicts. A player can state what his character is attempting to accomplish and, at the same time, specify some other goal that he, as a player, wants to see happen. The result can be that one, neither, or both goals are attained.

RPG Design Patterns Identified

Dice Pools, Idiom, Last Man Standing, Negotiated Contest, Point Spend Attribute, Recycled Fortune (Initiative and Conflict Resolution), Resource (Money), Trauma Gauge (Spunk)

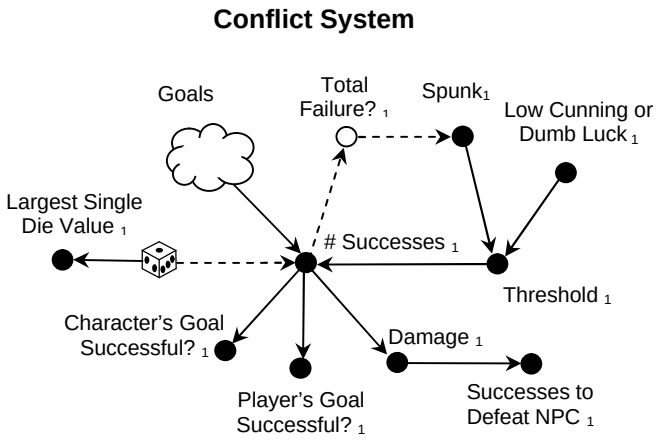
Character Makeup

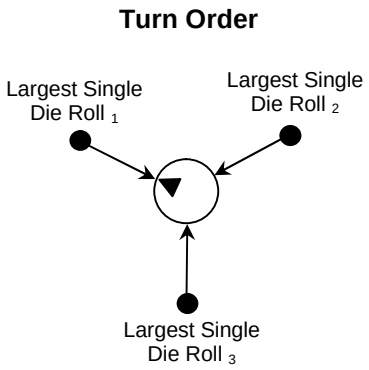
Characters have three core attributes: “Spunk,” “Dumb Luck,” and “Low Cunning.” Spunk is initially set to a value of 5 while the others are set to zero. Dumb Luck and Low Cunning can be raised by spending Spunk points. Each point of Spunk gives the player two points to distribute to the other attributes. Characters also have an idiom called “Stage” which can be set to “Oral,” “Anal,” or “Genital.” Oral characters are always on the lookout for food. Anal characters are fart factories. Genital characters are, well, depraved. Characters gather treasure in the form of magic items and “Coins,” which can be spent on nifty new equipment.



Conflict System

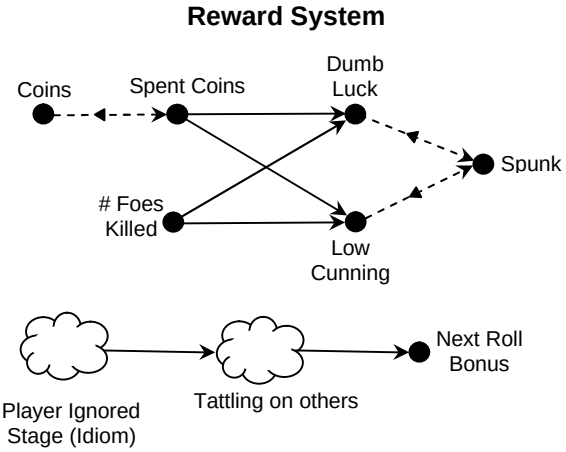
Elfs uses a dice pool system for contests consisting of 3d10. The individual dice rolls are compared to a threshold. Any values less than or equal to the threshold are counted as successes. Spunk is always counted into the threshold, but it may be raised by either Low Cunning or Dumb Luck, depending on what actions are attempted. Low Cunning is used on actions involving dirty, rotten tricks where the character takes advantage of someone else. Dumb Luck is used when the player states two desired outcomes to a conflict, one that he wants and one that his character wants. In such a roll, the player’s desired outcome happens if the roll comes up with any successes. The character’s wishes are fulfilled only if all three dice roll successes (total success). If a conflict results in total failure for a character (zero successes), the character sustains damage in the form of a one point loss of Spunk.





Turn Order

The order of character actions is done by first rolling the 3d10 for contests, taking the highest single die roll from all characters involved, and going in order from highest to lowest. Thus, the dice rolls are used for both action order and conflict resolution.



Reward System

Every 100 Coins spent (not just looted) and every 10 enemies slain earn a player the right to raise his character's Low Cunning or Dumb Luck by one point. Also, players are encouraged to "tattle" on the other players when they don't properly portray their characters' Idioms. Whenever this is done, the tattling player gets a bonus on his character's next roll.

Fudge

Fudge was written by Steffan O'Sullivan and is published by Grey Ghost Press, Inc. The game is best characterized by its flexibility. Virtually all aspects of Fudge can be customized. The game master has a couple of choices on what dice to roll in conflicts, and can even elect to go completely diceless. He also selects the genre in which the game will be played out (along with its setting) and chooses what attributes are appropriate for characters. The game text helps out in these endeavors by providing a plethora of possibilities. In the end, *Fudge* turns out to be more of a role-playing game toolkit than a pre-defined game. It is a framework onto which customized mechanics can be grafted. That might be great for game masters who enjoy tailoring systems to their own needs, but it could be inconvenient for anyone looking for something concrete that doesn't require a lot of thought before play begins.

RPG Design Patterns Identified

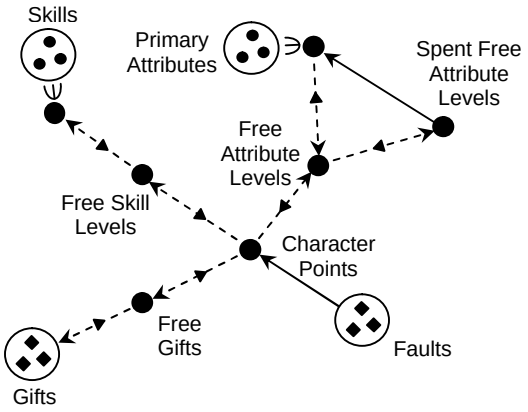
Dice Pool, Flaw, Game Master, Gift, Point-Spend Attribute, Generalized Contest, Random Attribute (optional), Rank, Template, Trait (Fudge calls them "skills," but they aren't pre-defined)

Character Makeup

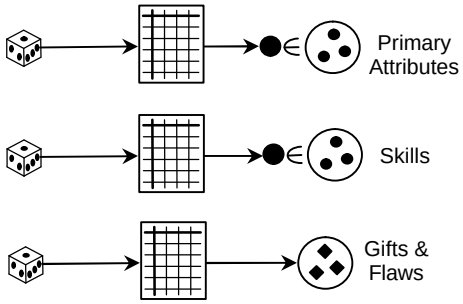
Fudge gauges have textual values which are often associated with numbers when used in conflicts: "Terrible" (-3), "Poor" (-2), "Mediocre" (-1), "Fair" (0), "Good" (+1), "Great" (+2), "Superb" (+3), and (sometimes) "Legendary" (+4). These textual values are used to describe one individual's characteristics in comparison to others of his "scale." So, humans would be rated on their own scale, while anthropomorphic bugs would be rated on their own scale. If a character of one scale conflicts with a character of another scale, then adjustments for scale must be taken into account. So, a human with "Mediocre" strength could out-lift a grasshopper of "Legendary" strength because the insect scale factor for strength would be more than 6 points less than the human scale factor. The point of the scaling factors is to make the vernacular and comparisons of similar character types easy, since most stories involve interactions between characters of a like nature.

The components that make up *Fudge* characters are so flexible that there really are not any attributes consistent from one game master to the next, although those that *are* used are assigned textual values as described above. The only gauges that even come close to being universal are "Damage Capacity" and "Wound Level." Damage Capacity has the effect of reducing the effects of wounds while Wound Level acts as both a form of Hit Points and a Trauma Gauge. And, even these characteristics can be dropped or radically modified if desired. The standard use for Damage Capacity is to subtract its associated numerical value from any delivered damage. Wound Levels are rated as "Just a scratch," "Hurt," "Very Hurt," "Incapacitated," "Near Death," and "Dead."

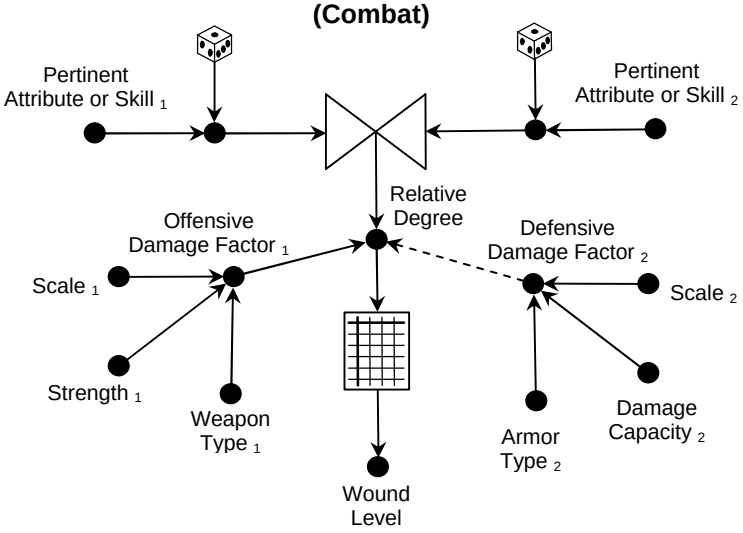
Character Makeup
(Objective Character Creation)



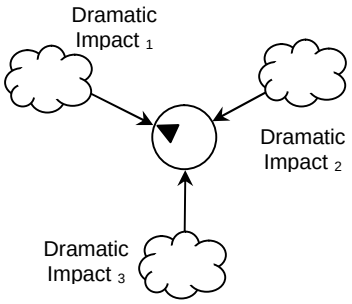
Character Makeup
(Random Character Creation)



Conflict System
(Combat)

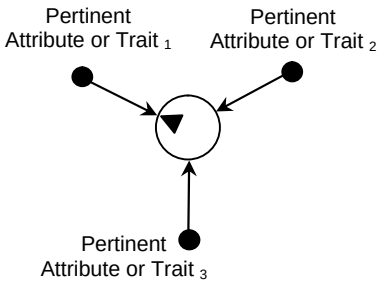


Turn Order
(Option 1)

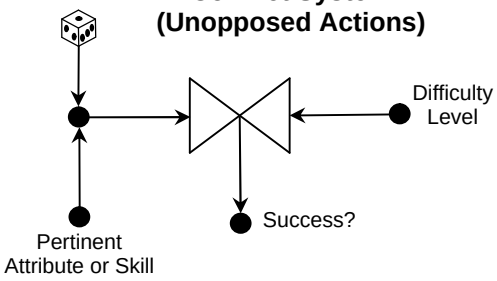


Note: In Option 2, all actions are simultaneous, so no diagram is needed for it.

Turn Order
(Option 3)



Conflict System
(Unopposed Actions)



One interesting option for handling Wound Levels splits out the various wound rating values out into gauges of their own, each of which is associated with its own detrimental effects. So, Just a Scratch becomes a gauge, Hurt becomes another, and so on. Each of these gauges is assigned a maximum value, and any damage that exceeds that maximum spills over into the next higher wound gauge. For example, the Just a Scratch gauge may have a maximum of 3. So, the character can sustain 3 scratches and these would be used to increase the value of the Just a Scratch gauge. However, the 4th scratch would overflow the capacity of the Just a Scratch gauge, causing the scratch to spill over into the Hurt gauge. The Hurt gauge, in turn, might have a maximum of 2. So, if the character takes more than 2 Hurt wounds, they would spill over into the Very Hurt gauge, and so on. One interesting feature of this technique is that a character may sustain a single Very Hurt wound and suffer the consequences of that injury but have no Just a Scratch or Hurt wounds at all.

Fudge provides a few suggestions on how characters should be generated:

1) Objective Character Creation starts by setting all attributes to a default value. A player is then given character points ("Free Levels") to spend in raising some of these values. He also has the option of lowering some attribute values to earn more points that he can spend elsewhere.
2) Alternately, the game master can have players generate their characters randomly. In this technique, players roll 2d6 and use the generated number to set their character's values via a table lookup.
3) A Subjective Character Creation method is also available, if the game master so chooses. In this option, players essentially decide how their character's traits and values are set. The game master might provide guidance in this endeavor by allowing a character to have a "Superb" rating in a small number of attributes, "Great" in a few more, and "Good" in the remainder.

Fudge encourages game masters to consider giving their players a resource known as "Fudge Points". Fudge Points can be spent to modify dice rolls, lower the severity of wounds, and introduce convenient facts into the storyline. In essence, Fudge Points provide a means to give players other than the game master some authorial control over the game world independent of their characters.

Conflict System

At its most basic level, *Fudge* resolves contests by comparing values derived from character gauge values (such as attributes or skills), dice rolls, and other factors. How those derived values are determined varies from one Fudge group to the next, since *Fudge* strives to remain imminently customizable. The text does provide some suggestions on how to resolve conflicts, though.

First, a value is generated in some way. Some of the suggested techniques are

1) Generate a random value by rolling four "Fudge Dice." A Fudge die is a cubical die with a "+" on two sides, a blank on two sides, and a "-" on the remaining two sides. Thus, each die roll results in a "+1," "0," or "-1" value. These are added up to give a range of -4 to +4.
2) Generate a random value by rolling 3d6, add the dice together, and perform a table look up to come up with a value in the range of -4 to +4.
3) Avoid rolling dice altogether. Instead, rely on players either expending Fudge Points to augment their chances in contests or have them narrate some reasonable way in which their characters improve their chances in conflicts.

However the augmenting value is generated, add it to the numerical value associated with the characteristic that is most pertinent to the task at hand, be it the character's "Strength," his "Great Sword" skill, or whatever. This results in a final value to compare against an opposing value.

For actions against opponents, the opposing value is generated in the same way, using that character's most pertinent gauge value. For unopposed actions, the game master sets the difficulty level that must be overcome. The difference between the two final values is known as the "relative degree." Damage is determined by adding the winner's "relative degree" to his "Offensive Damage Factor" and then subtracting the opponent's "Defensive Damage Factor." The "Offensive Damage Factor" incorporates the character's strength, the type of weapon, and scale. The "Defensive Damage Factor" incorporates the defender's "Damage Capacity," armor, and scale. The resulting damage value is converted into a wound level by a table lookup.

Turn Order

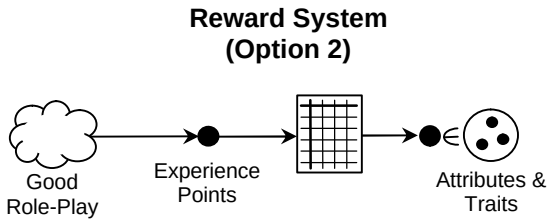
True to form, *Fudge* gives the game master several options on how to order character actions in extended conflicts:

1) Pace actions according to "story elements." In other words, the game master directs the timing of actions based on their dramatic impact.
2) Make all opposed actions simultaneous. That is, when characters come into conflict with one another, have each side roll dice as normal to gauge their overall success in that combat round. The winner inflicts damage on the loser.
3) Separate offensive and defensive rolls. In this option, the game master must choose an attribute or trait to use in determining who acts first. For example, he could give all characters an "Initiative." "Speed." or "Agility" attribute for this purpose. Characters perform actions in an order based on this gauge going from highest value to lowest. Alternately, the game master could just decide to roll dice to determine the action order randomly taking into account modifiers for the environment and/or character abilities.

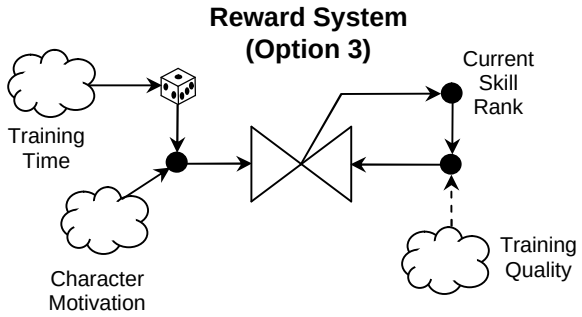
Reward System

Fudge just wouldn't be *Fudge* if it provided only one way to reward players. Some of the possibilities it suggests are

1) Have the game master reward players on a purely subjective basis. That is, characters advance only when the game master decides it would be dramatically appropriate for the character to gain a rank in some skill or attribute.
2) Award players experience points based on their quality of role-play. Experience points can then be spent on raising attribute values and trait ranks. The costs for advancement are based on a table lookup.



3) Allow players to have their characters get training when they want to raise a trait rank. The character's current rank and the quality and length of training determine the difficulty of a roll that must be made for the character to improve.



GURPS (Third Edition, Revised)

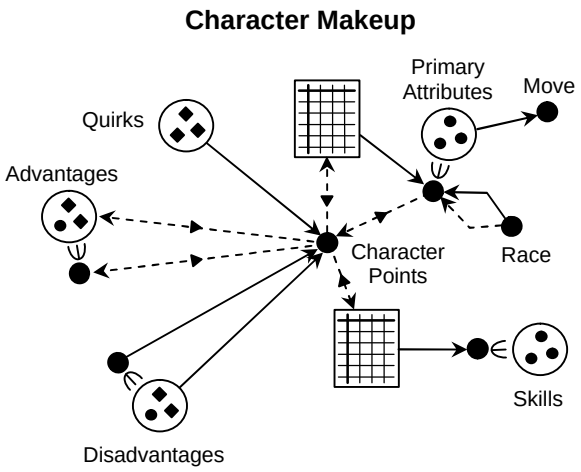
GURPS (Generic Universal Role-Playing System) is published by Steve Jackson Games. True to its name, it is a role-playing framework independent of any particular setting or genre. That is not to say that there aren't many supplements available to provide these details, though, both generic and specific fictional settings. It is just that the system itself is not geared toward any one in particular. *GURPS* supplements span the spectrum from low-tech fantasy to high-tech sci-fi along with just about any combination in-between.

RPG Design Patterns Identified

Damage Resistance, Game Master, Generalized Contest, Hit Points, Last Man Standing, Point Spend Attribute, Rank, Skill Tree, Success Reward, Trauma Gauge

Character Makeup

GURPS characters are designed using a system based purely on "Character Points." Character Points are a resource that players spend to customize their characters. Different "Skills" and "Advantages" cost varying amounts of Character Points depending on their difficulty and overall usefulness. Additional Character Points can be gained by accepting various "Disadvantages" and "Quirks." Characters have four primary attributes of "Strength," "Dexterity," "Intelligence," and "Health" whose values are adjusted based on the character's chosen race. These attributes usually range in value between 1 and 18, but can extend even higher if a player is willing to sacrifice elsewhere. Their values are set depending on the number of Character Points spent and the costs are based on a table lookup. Unusually low scores earn a player more points to spend elsewhere. Characters also have "Hit Points," which are based on the Health attribute.

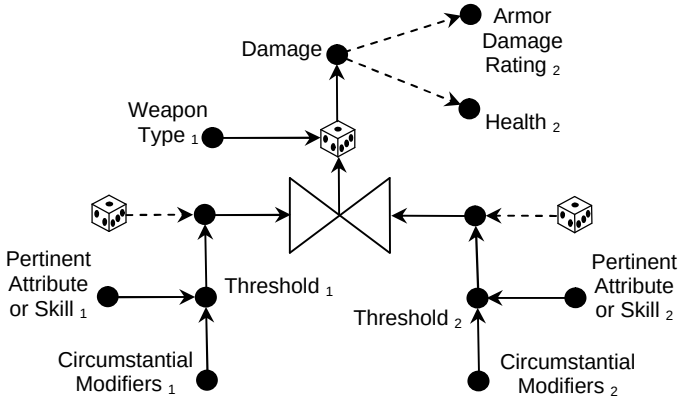


Skills are ranked. Like everything else in the game, skill ranks are determined by how many Character Points are expended on the skill. The rating also factors in both the skill "Difficulty" and the primary attribute of Dexterity or Intelligence, depending on whether the skill is physical or mental in nature. Although the skill rank, or "Level," takes into account all these various components, the rating is essentially a primary attribute added to a skill adjustment purchased with Character Points.

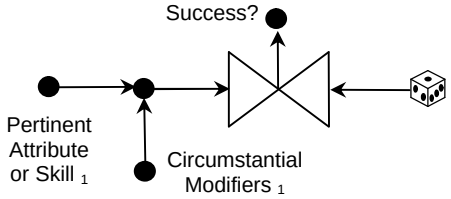
Characters also have a "Move" attribute that is derived from Health and Dexterity. This score represents the number of yards a character can run in a second. Characters have a

number of other derived attributes as well: “Dodge,” “Parry,” “Fatigue,” “Block,” “Damage Resistance,” “Encumbrance,” etc.

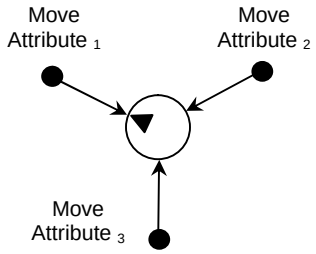
Conflict System (Combat)



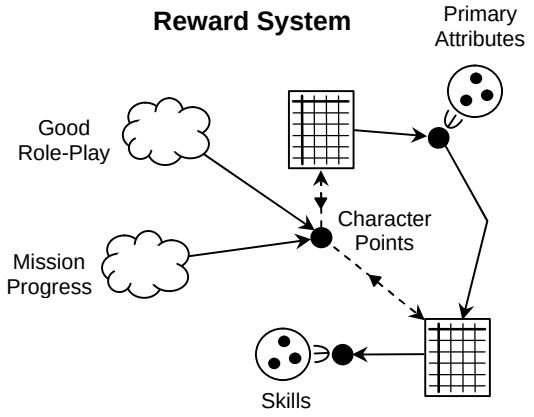
Conflict System (Unopposed)



Turn Order



Reward System



Conflict System

Conflicts are resolved by rolling 3d6, summing the numbers, and then comparing the total to a set threshold. The threshold generally equals a character's pertinent attribute or skill rank, but may be adjusted due to circumstances. If the rolled sum is less than or equal to the threshold, the character succeeds. Otherwise, he fails.

If two characters compete against one another, the resolution technique is different based on whether the contest is "quick" or "regular." A quick contest is one that is over is about a second. Both sides of the contest roll as above. The winner is the character who succeeded by the most or failed by the least. If the roll results in a tie, neither side wins. Regular contests take more time. If one side succeeds on his roll and the other fails, the contest goes to the winner. If both fail or both succeed, the contest is not yet decided and both sides must re-roll until one side is victorious.

For conflicts involving combat, *GURPS* is a traditional "roll-to-hit, roll-to-damage" system. If an attack hits, dice are rolled to determine how much damage is inflicted. This value can be altered by armor via its "Damage Rating," which is subtracted from the rolled damage. The remainder is subtracted from the character's hit points (Health).

Turn Order

The order of combat actions is purely Karma-based. The Move attributes of characters are compared. Those with higher Move scores go first. Once everyone has gotten his turn, the cycle repeats until the conflict ends.

Reward System

Since *GURPS* is entirely based on Character Points, it is no surprise that the game's sole reward system consists of the game master awarding these to players. This is done at the end of every gaming session and represents a reward for "Good Play." The award incorporates the GM's judgment of the quality of role-play and his assessment of the characters' progress toward their stated missions. The amount awarded can range anywhere from -5 to 5 points per character, so the "reward" can actually be a penalty if the players do a particularly poor job.

HARP (High Adventure Role Playing)

HARP is published by Iron Crown Enterprises. It is a traditional skill-based fantasy game that is essentially a simplified and abbreviated version of *Rolemaster Fantasy Role Playing* (also published by ICE).

RPG Design Patterns Identified

Class, Experience Points, Game Master, Generalized Contest, Gifts ("Talents"), Hit Points, Last Man Standing, Level, Point Spend Gauge, Race, Random Attribute, Rank, Skill

Character Makeup

HARP characters have eight primary attributes ("stats") of "Strength," "Constitution," "Agility," "Quickness," "Self-Discipline," "Reasoning," "Insight," and "Presence." These attributes range in value from 1 to 105. The values are set by one of three options:

1) Roll a d100 eight times, re-rolling any value less than 40 and assign the numbers to the attributes however desired.
2) Use a point-spend attribute system. In this case, every character starts with 550 points to spend on his attributes. The cost to obtain a given score is provided via a table lookup.
3) Use the method described in (2), but instead of giving every character 550 points, give the character a number of points equal to 500 + 10d10.

The final attribute values generated gives the player an initial number of "Development Points" with which to further customize his character.

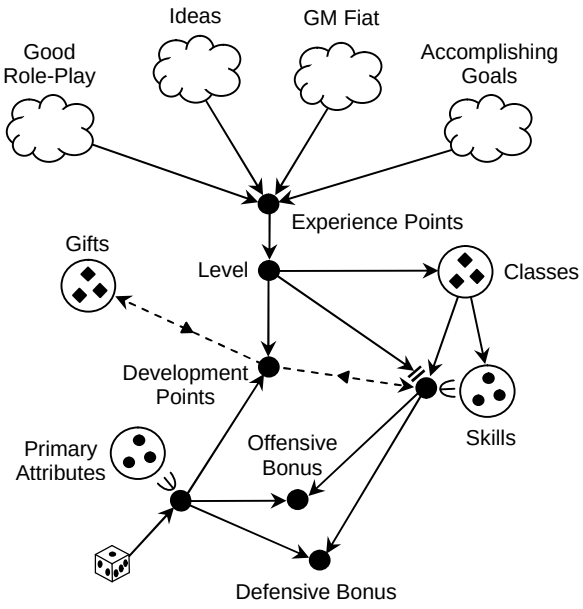
Every *HARP* character has a profession (see the Class Pattern) of one of the following: "Cleric," "Fighter," "Harper," "Mage," "Monk," "Ranger," "Rogue," "Thief," and "Warrior Mage." Each of these classes provides the character with a group of "favored skills" along with an initial starting rank for those skills. Gaining further ranks in a favored skill costs 2 Development Points per rank. Gaining ranks in a non-favored skill costs 4 Development Points per rank. Skill ranks cannot exceed a value equal to 3 times the character's level plus 3 (see the Level Pattern).

Characters also have both a race and a culture which provide adjustments to the primary attributes. A character's race is selected from the following list: "Human," "Elf," "Dwarf," "Gnome," "Halfling," and "Gryx." The available cultures are: "Dwarven," "Gnomish," "Sylvan," "Nomadic," "Rural," "Urban," and "Halfling."

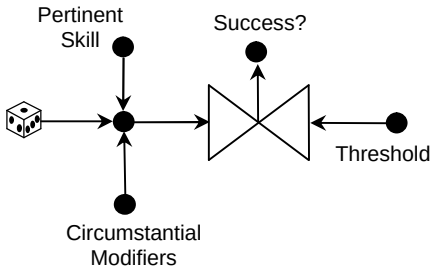
For combat situations, characters have an "Offensive Bonus" and a "Defensive Bonus." The Offensive Bonus is comprised of the skill rank bonus of the weapon being used, adjustments for Strength and Agility, bonuses from any pertinent gifts, magical adjustments for weapons, and other environmental modifiers. The Defensive Bonus is

derived from the character's Quickness, armor, pertinent gifts, magical adjustments, and other environmental factors.

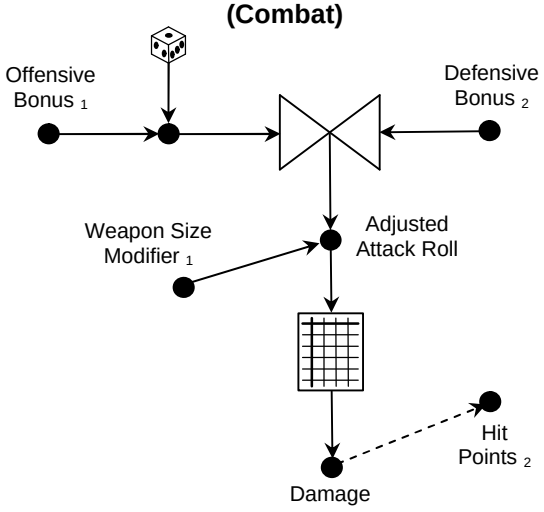
Character Makeup & Reward System



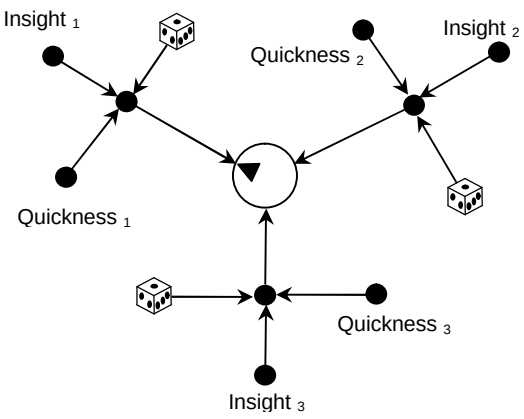
Conflict System (Unopposed)



Conflict System (Combat)



Turn Order



Conflict System

Combat attacks and maneuvers are performed by first making an "open-ended" d100 roll. The roll is called "open-ended" because any roll resulting in a value of 96 to 100 allows the player to re-roll the dice and sum the results. As long as the subsequent rolls fall into the range of 96 to 100, this process continues and the results accumulate. If the roll falls into the weapon's "fumble range", then the player makes another roll and consults a fumble table for effects.

Maneuvers (the use of skills) are made by making an open-ended d100 roll, adding the character's bonus in the pertinent skill, and modifying it appropriately with environmental adjustments. If the result is 100 or greater, the character succeeds at his task.

Combat attacks are made by performing an open-ended d100 roll, adding the attacker's Offensive Bonus, and subtracting the target's Defensive Bonus. The attack lands if this "Total Attack Roll" equals 1 or greater. In this case, the size modifier of the weapon is added to derive the "Adjusted Attack Roll." This value is used in a table lookup to determine the damage delivered to the target, which may include both a hit point deduction along with other traumatic effects (such as stun, bleeding, or other combat penalties).

Turn Order

Players declare actions before rolling a d10 for initiative. To the rolled d10 value are added adjustments for the character's Quickness and Insight attributes along with environmental modifiers. Characters go in order of the highest sum to the lowest. This process is repeated at the beginning of every combat round, each of which is assumed to last for 2 seconds.

Reward System

HARP rewards players through experience points. The total accumulated amount of experience points determines the character's level (see the Level Pattern). This, in turn, determines the number of Development Points the character has to spend on skill ranks and additional classes and gifts. Experience points are awarded for good role-playing, good ideas, and anything else the GM wants to award. The largest experience point awards, though, are given for accomplishing goals. Goals can be either major or minor and can be either party or personal. In general, major goals are worth more than minor goals and party goals are worth more than personal goals. The number of experience points awarded is subjectively based on the game master's assessment of the goal's difficulty.

HeroQuest

HeroQuest was written by Robin D. Laws and Greg Stafford and is published by Issaries, Inc. It is a fantasy game set in the world of Glorantha during the "Hero Wars." Created by Greg Stafford, Glorantha is a rich fantasy world originally conceived in 1966 before RPG's even existed. It has evolved over the years attracting many writers and gamers in the process. Glorantha was used as the setting for *RuneQuest*, but was divorced from that game after Chaosium sold *RuneQuest* to Avalon Hill. (Avalon Hill was eventually taken over by TSR, which was sold to Wizards of the Coast along with the rights to D&D. WotC was eventually acquired by Hasbro. Needless to say, Hasbro decided D&D didn't need the competition and *RuneQuest* died an unfortunate death.) *HeroQuest* actually came much later than Glorantha, is more free-form in its design than *RuneQuest*, and is tailored specifically to the world.

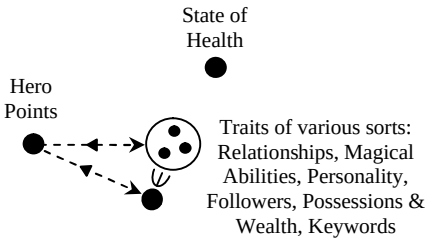
RPG Design Patterns Identified

Conflicted Gauge (wound traits can be used to aid success in some kinds of conflicts even though they are most easily conceptualized as penalties), Currency, Game Master, Negotiated Contest, Race, Rank, Resource ("Hero Points"), Template, Trait, Trauma Gauge

Character Makeup

HeroQuest characters are almost entirely trait based. All characters possess a resource known as "Hero Points" that are spent in raising ranks in the character's various traits. It can also be expended to improve the outcomes of conflicts in the hero's favor.

Character Makeup



All characters do have a "State of Health," though, that follows the Trauma Gauge design pattern. The trait can take on one of the following states: "Healthy," "Hurt," "Impaired," "Injured," "Dying," or "Dead." However, this is not a strict ranking, because a character can have multiple "Hurts," each of which has a detrimental effect on the character's capabilities.

The remaining character traits fall into the six categories of "Skills," "Relationships," "Magical Abilities," "Personality," "Followers," and "Possessions & Wealth." (To avoid confusion, please note that the "Skills" category is really a list of trait abilities and does not conform to this book's definition of the term "skill.") Characters also have a number of "Keywords." Keywords are especially important in that they convey a character's "Homeland," "Occupation," "Magic," and "Common Magic." Keywords are very broad ranked traits which incorporate everything associated with the culture or profession specified. Here we call Keywords "traits" rather than "skills," even though there are pre-defined write-ups of them in the rulebook. Although these write-ups are often used directly, they really only comprise an extensive list of examples. That is, players are encouraged to create their own as well. So, players either make up their own Keywords or select them from a list of examples. (You could rightly claim that

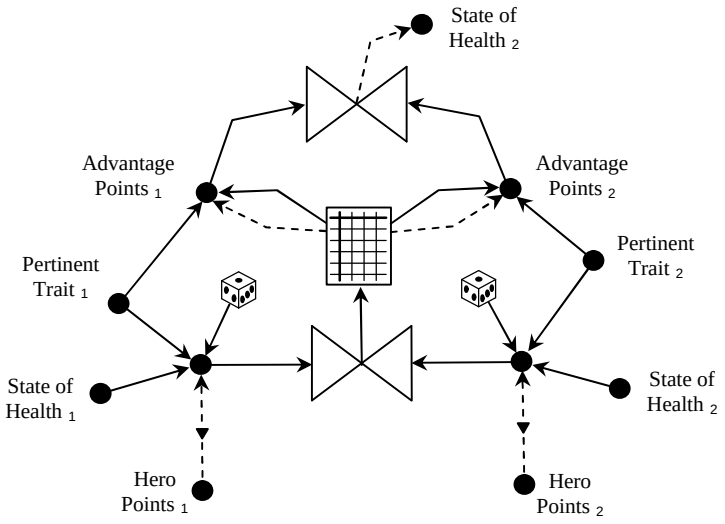
HQ Keywords fall halfway between this book's definition of "skill" and "trait.") Even though the Keywords have ranks, they also act as templates, giving lists of suggested traits that characters with the keyword should commonly adopt. For example, the Occupation of "Hunter" suggests characters take the "Archery," "Track," and "Wilderness Survival" traits, among others.

Traits are ranked but the ranking scheme is not a simple numeric value. Rather, each trait has a numeric value and a "mastery" value. Twenty is the largest numeric value allowed. If the rank is increased to 21, it automatically drops to 1 and the mastery value associated with the rank increases by 1. Masteries are denoted by the runic symbol ᛝ in *HeroQuest*. So, a rank 13 trait with 2 masteries is denoted as 13 ᛝ 2.

Conflict System

Conflicts are handled as either "simple contests" or "extended contests." A simple contest is one which takes only moments to resolve and do not necessarily deal with the primary purpose of the campaign. Simple contests are handled with a single die roll. Extended contests, on the other hand, are conflicts that take more time to resolve. They are essentially broken down into a number of smaller conflicts and commonly require several die rolls to determine a victor.

Conflict System (Extended Contests)

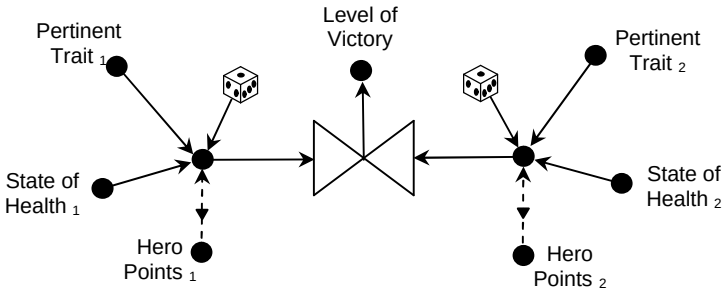


HeroQuest uses opposed d20 rolls to determine success in conflicts. Each d20 roll is compared to a threshold (determined by the rank of the skill being used by each side). If the die roll comes up as a 1, the action succeeds brilliantly. If it comes up greater than one but less than the threshold, the character succeeds normally. If the result is greater than the target number but less than 20, the action fails. And, finally, if the die rolls a 20 the character fails horribly. These four degrees of success and failure are labeled "Critical," "Success," "Failure," and "Fumble."

These degrees of success and failure can be "Bumped" up or down in two ways. First, if the character has masteries in the trait being used greater than that of his opponent, he can bump the success result up by the difference in masteries. Second, Hero Points can be used to bump the result. Either way, a Failure can be bumped up to a Success or a Success can be bumped up to a Critical. If a character has masteries "left over" after

bumping his success up to Critical, he can apply the remaining masteries to bump down his opponent's success.

Conflict System (Simple Contests)



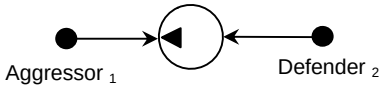
The resulting degrees of success and failure of both sides are compared. Depending on the difference in these outcomes, the overall "level of victory" of the winner is: "Tied," "Marginal," "Minor," Major," or "Complete."

In extended contests, the ongoing success of each side is rated in the form of "Advantage Points," or "APs." These are initially set based on the rank of the trait being applied as the first action of the conflict. The values wax and wane as each side makes gains or suffers losses in the contest. Before each roll, each side makes an AP bid. Depending on the degree of success of the winner on every roll, the AP bid is multiplied by a value. Depending on both sides the degrees of success in their rolls, the AP is then either merely subtracted from the loser's AP or actually transferred from the loser to the winner. When either side's AP total drops to zero or less, the contest is over. The end result of the contest depends on the loser's final AP total as determined by a table lookup. For combat, a character's State of Health can be altered to Hurt, Impaired, Injured, or Dying.

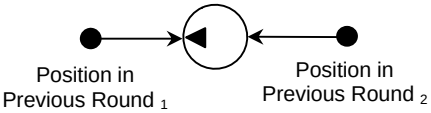
Turn Order

On the first action of an extended contest, the aggressor's action goes first followed by his opponent's. If the contest is not over (neither side's AP total has dropped to zero or less), a new round begins with each person taking his turn in a round robin fashion. So, in general, the same character takes the first action on each round.

Turn Order (first round)



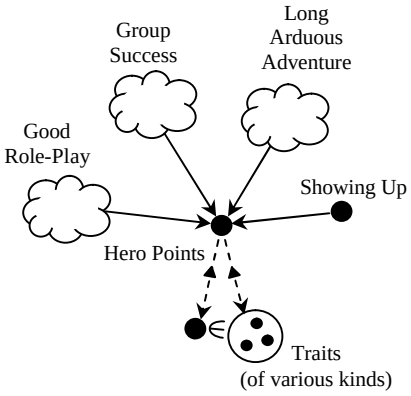
Turn Order (subsequent rounds)



Reward System

HeroQuest rewards players through the distribution of Hero Points. At the start of every adventure, each character gains 1 to 5 points. A long and arduous adventure lasting several sessions will earn a character another 1 to 5 points, depending on the group's overall success and on the quality of role-play.

Reward System



Hero System 5th Edition

Hero System 5th Edition was written by Steven S. Long and is published by DOJ, Inc. It is the descendant of *Champions*, a super-heroes RPG first published in 1981 by Hero Games. Touting itself as "The Ultimate Gamer's Toolkit," Hero System is designed to be appropriate for "ordinary" heroic adventures as well as super-hero style play. At first glance, the daunting 372 page manual might be mistaken for a textbook rather than a game book. Its pages contain a vast array of pre-defined abilities that can be tailored to a player's character concept. The system allows characters to be highly customized, allowing each to have unique abilities, equipment, and powers.

RPG Design Patterns Identified

Flaws ("Disadvantages"), Game Master, Generalized Contest, Gifts ("Talents" and "Perks"), Hit Points ("Body"), Point Spend Attribute, Rank, Skills ("Skills" and "Powers"), Success Reward ("Character Points"), Template

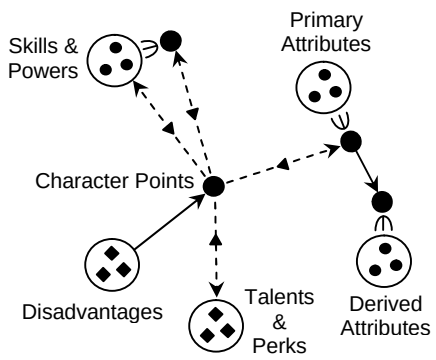
Character Makeup

Hero characters have eight primary attributes ("Characteristics"): "Strength," "Dexterity," "Constitution," "Body," "Intelligence," "Ego," "Presence," and "Comeliness." Their values are specified by having players spend "Character Points" and the costs vary from attribute to attribute. The genre of game, whether heroic or super-heroic, determines the number of points distributed to players to generate their characters. After setting the primary attribute values, players derive six more attributes from the primary ones via various formulae. These derived attributes are: "Physical Defense," "Energy Defense," "Speed," "Recovery," "Endurance," and "Stun."

Body and "Stun" are both hit points attributes. Characters fall unconscious when Stun drops to zero. They start dying when Body falls to 0 and truly die when it drops to -10. Characters also possess two additional derived attributes known as "Combat Value" and "Ego Combat Value," which gauge character effectiveness in combat and mental actions, respectively. Combat Value is further adjusted by various skill ranks, equipment, and other factors to produce both an "Offensive Combat Value" and a "Defensive Combat Value."

Players purchase skills (Skills and Powers) and gifts (Perks and Talents) for their characters using Character Points. Character Points also buy skill ranks at costs that vary from skill to skill. Players can earn additional Character Points by accepting flaws (Disadvantages) for their characters.

Character Makeup



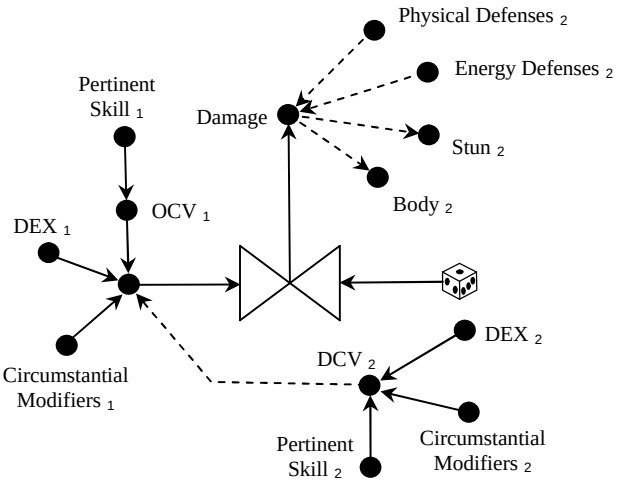
Powers are particularly notable, in that they are skills that essentially define mechanical

effects without specifying "special effects." For example, the game provides an "Energy Blast" power. A player may customize this power to fit his character concept by describing it as the power to cast thundering lightning bolts from his fingertips, summon blazing spheres of greenish flames from the heavens, or shoot laser beams from a blaster. Essentially, all Energy Blast powers have the same mechanical effect. Their differences mainly involve presentation. So, players craft powers to match their character concepts rather than the other way around. Despite their flexibility, hero powers qualify as "skills" rather than "traits" because the various powers are pre-defined, even though their superficial appearance is entirely in the players' control.

Conflict System

To perform a combat attack, a player rolls 3d6 and sums the result. A blow lands if the sum is less than or equal to a value determined by both the attacker's Offensive Combat Value and the defender's Defensive Combat Value (11 + OCV – DCV). Mental attacks use a similar technique, replacing Ego Combat Value for "Combat Value.

Conflict System (Combat)

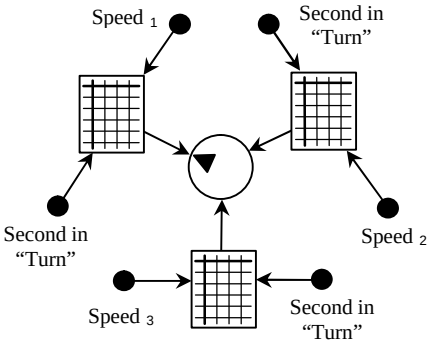


If an attack lands, it inflicts damage on the target. Depending on the nature of the attack, the damage may be subtracted from either Stun or Body (or both). Stun attacks can knock an opponent out, but cannot kill. Body attacks can kill. Physical Defenses (such as armor) and Energy Defenses (such as force fields) can modify the amount of damage actually sustained by a character from an attack.

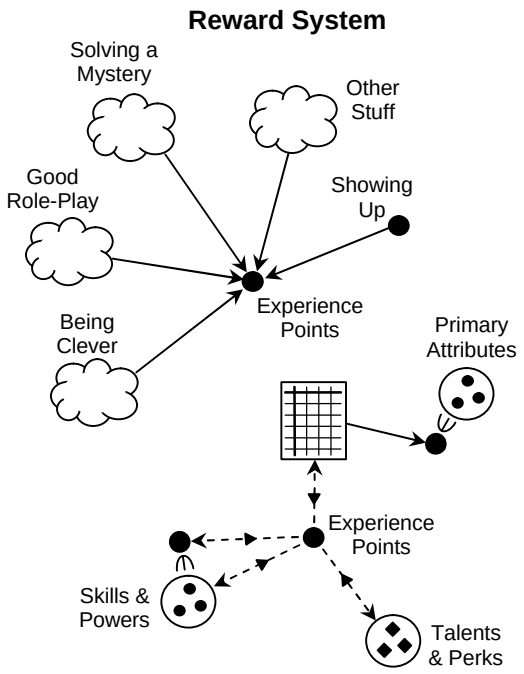
Turn Order

Combat is segmented into 12 second "turns." A character's Speed attribute determines number and timing of his combat actions every turn. If a character has a 3 Speed, he gets three actions ("Phases") every turn. The timing of those actions is determined by a table lookup. For example, a character with a 3 Speed performs his actions on 4th, 8th, and 12th second of every turn since that is what the table specifies. For characters acting on the same second, the action order goes from highest Dexterity to lowest.

Turn Order



Some actions count as “Full Moves” while others count as “Half Moves.” In general, a character can perform either one Full Move or two Half Moves during each combat “Phase.” If a character performs a half move before an attack, then the attack counts as a half move. But, a character cannot perform an attack and then enact a half move. Once a character makes an attack, his Phase is over.



Reward System

The game master rewards players by giving their characters experience points, which are spent just like Character Points. Each session is generally worth at least 1 experience point, but players can earn more by being clever, role-playing well, solving a mystery, etc.

InSpectres

InSpectres was written by Jared A Sorensen and is published by Memento Mori Theatricks. The game is quite honest about its blatant adoption of ideas from the GhostBuster movies. In the game, the characters set up an InSpectres franchise that specializes in *"Battling the Forces of Darkness So You Don't Have To."* The setting is modern day San Francisco with the exception that ghosts and spooks are common, everyday pests. The characters (agents) are exterminators.

Stories are strictly structured: A client contacts the franchise to deal with some paranormal menace; the agents investigate the problem through interviews and other research; the characters beg, borrow, or otherwise acquire the gear needed to defeat the bad guys before heading out into the field; the agents wrangle with the nasties, defeating them in whatever fashion presents itself; the franchise collects its fee; and the characters take a vacation until the next client shows up.

InSpectres noticeably lacks any form of Hit Points. Characters simply cannot die from their ordeals unless a player decides that his character's death would be entertaining. So, players are free to hose their characters in any variety of comical ways.

One unique concept in *InSpectres* is known as a "Confessional." This is a point where a player has his character talk directly to the other players. Not the other *characters*, mind you, the other *players*. It is equivalent to an actor in a movie looking directly to the camera and addressing the audience. The Confessional allows a player to foreshadow events that he would like to see in upcoming scenes or to introduce new information or plot twists that the players, but not necessarily the characters know about: "As the crew fled from the cackling banshee down isle four of the FoodMart, little did we realize how crucial pickled eggs would be to our salvation."

RPG Design Patterns Identified

Attribute, Dice Pools, Endgame, Faction (Heroes vs. spooks), Game Master, Narrative Reward, Negotiated Contest, Point Spend Gauge, Resource, Shared Gauge ("Library Card," "Gym Card," "Credit Card," "Bank"), Shared Power, Structured Story, Traits, Trauma Gauge (stress reduces the effectiveness of attributes)

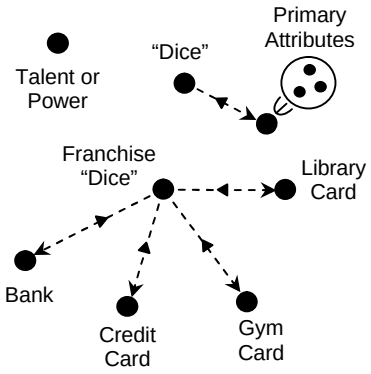
Character Makeup

InSpectres characters have four primary attributes: "Academics," "Athletics," "Technology," and "Contact." "Normal" characters (i.e., humans) are given 9 dice to distribute to these four attributes, but all must lie in the range of 1 to 4. "Weird" characters (i.e., werewolves, vampires, etc.) are given 10 dice to spend, with each attribute value falling into the range of 0 to 10. The game allows only one weird agent per game.

There are two additional attributes of "Current Stress" and "Current Cool." Cool has a range of 0 to 3 for "Normal" characters, but is unlimited for "Weird" characters.

Each "Normal" character is given a unique trait ("Talent") by his player that makes him special. These can be as wide ranging as "Airplane Mechanic," "Baseball Card Collector," "Trivial Pursuit guru," "Green Beret," or "Dr. Who Fan." "Weird" characters lack this benefit. They do, however, get "Powers." (A vampire might be able to turn into a bat, for example.)

Character Makeup



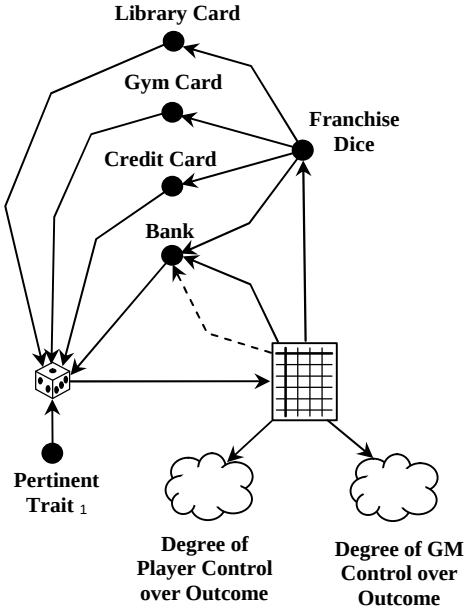
One other very important aspect of the character makeup is not actually part of the character at all. The InSpectres franchise has its own attributes that are shared among all of the franchise members. These are: Library Card, Gym Card, and Credit Card. These can be used by any of the franchise members to augment rolls dealing with Academics, Athletics, and Technology, respectively. Beginning franchises start with 5 dice to distribute among the three cards. Any leftovers go into the Bank attribute. Dice in the bank can be used to augment any roll, but are sometimes inaccessible.

Franchise cards are really the only attributes in the game that improve with time. As the franchise prospers, all of its agents prosper along with it. As the franchise wanes, so too do its employees.

Conflict System

All conflicts use a number of d6s equal to the value of whatever attribute is most pertinent. So, if the conflict requires physical exertion, a character's Athletics attribute would most likely be the pertinent score. The dice are rolled and the highest single number rolled on any dice is used in a table lookup to determine the outcome. In general, the higher the outcome, the more control the player has over describing the outcome. Low rolls give the game master narrative control. If the highest number is a 5 or 6, the group earns franchise dice, which pushes the story closer to the endgame. Also, note that the conflict roll does not pit the player characters' skills against those of the monsters. Monster capabilities don't really enter into the picture in any mechanical way.

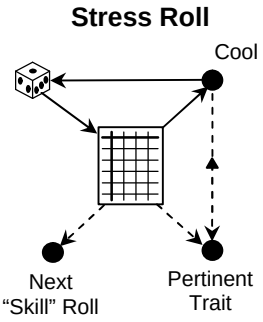
Conflict & Reward System



The conflict roll can be augmented with dice from the Bank or from the franchise cards, whichever is pertinent to the task at hand. These dice are "spent." But, depending on

the roll of the actual Bank dice (which are distinguished by a different color), you may actually get the dice back, lose an additional die, lose all of the remaining dice in the Bank, or even earn more dice to add to the Bank. The result is determined by a table lookup.

Characters can assist one another in conflicts. To do so, they must first state that they are assisting a comrade prior to rolling any dice for their own actions. Once the dice are rolled, the assisting player must hand over one of the dice he rolled to the player he is helping.



Anytime the game master deems it appropriate, he can call for a "Stress Roll." This is a special kind of conflict roll where the player rolls a number of d6, takes the lowest number, and performs a table lookup to determine the mechanical effects. Cool points mitigate the effects of stress. For every point of Cool, a character ignores one dice in any Stress Roll after the dice are rolled. So, a character with 2 points of Cool only has to consider the "best" die in any 3 dice Stress Roll. Franchise cards and other players are unable to assist Stress Rolls. Possible outcomes include gaining a point of Cool, suffering penalties on future conflict rolls, or losing points from various attributes. The causes of these mechanical effects are up to the players themselves. Lost trait points are regained at the end of each mission (when the character takes a vacation) or by spending Cool points.

Turn Order

InSpectres has no explicit rules governing the order in which players take turns. The conflict resolution system is set up in such a way that this isn't really necessary. A high conflict roll allows the player to describe what he'd like to see while a low roll indicates the game master gets that pleasure.

Reward System

The game rewards players with "Franchise Dice," which also works as an Endgame mechanic. At the beginning of every mission, the game master decides on how many franchise dice the story is worth. (The rulebook provides guidelines on what is reasonable.) Throughout the adventure, players earn these franchise dice through conflict rolls. When the pre-specified number of franchise dice has been earned, the game master declares victory and gives the dice to the franchise. Franchise dice augment the group's Library Card, Gym Card, Credit Card, and Bank attributes. The episode ends with the characters taking a well-earned vacation.

My Life with Master

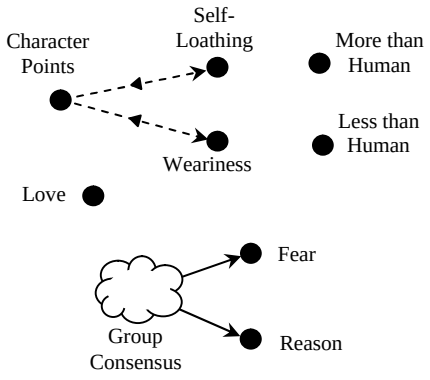
My Life with Master was written by Paul Czege and is published by Half Meme Press. In it, players portray potent but psychologically dependant minions of twisted masters who send them off to perform any number of inhuman tasks. Not only do the players design their characters, they also collectively design their master. The more the minions serve their master's diabolical interests, the more the minions hate themselves. Their increasing self-loathing makes them more and more capable of carrying out their master's orders, but at the same time strengthens their master's psychological grip on their psyches. A minion's only link to his human roots are his "Connections," which are people in town whom the minion cares about and visits on side trips whenever the master sends him out. The more a minion visits his Connections, the more Love he accumulates, which helps the minion resist the master's powerful hold. The goal of the game is for the minions to finally resist and kill their master.

RPG Design Patterns Identified

Attribute, Conflicted Gauge ("Fear," "Reason," "Self Loathing"), Dice Pool, Endgame, Flaw ("Less than Human"), Game Master, Gift ("More than Human"), Idiom ("Love"), Narrative Reward ("Intimacy"/"Desperation"/ "Sincerity"), Negotiated Contest (which is very coarse grained – one roll per scene), Safety Valve, Shared Gauge ("Fear" and "Reason"), Structured Story, Trait (More than Human/Less than Human), Trauma Gauge ("Weariness")

Character Makeup

Character Makeup

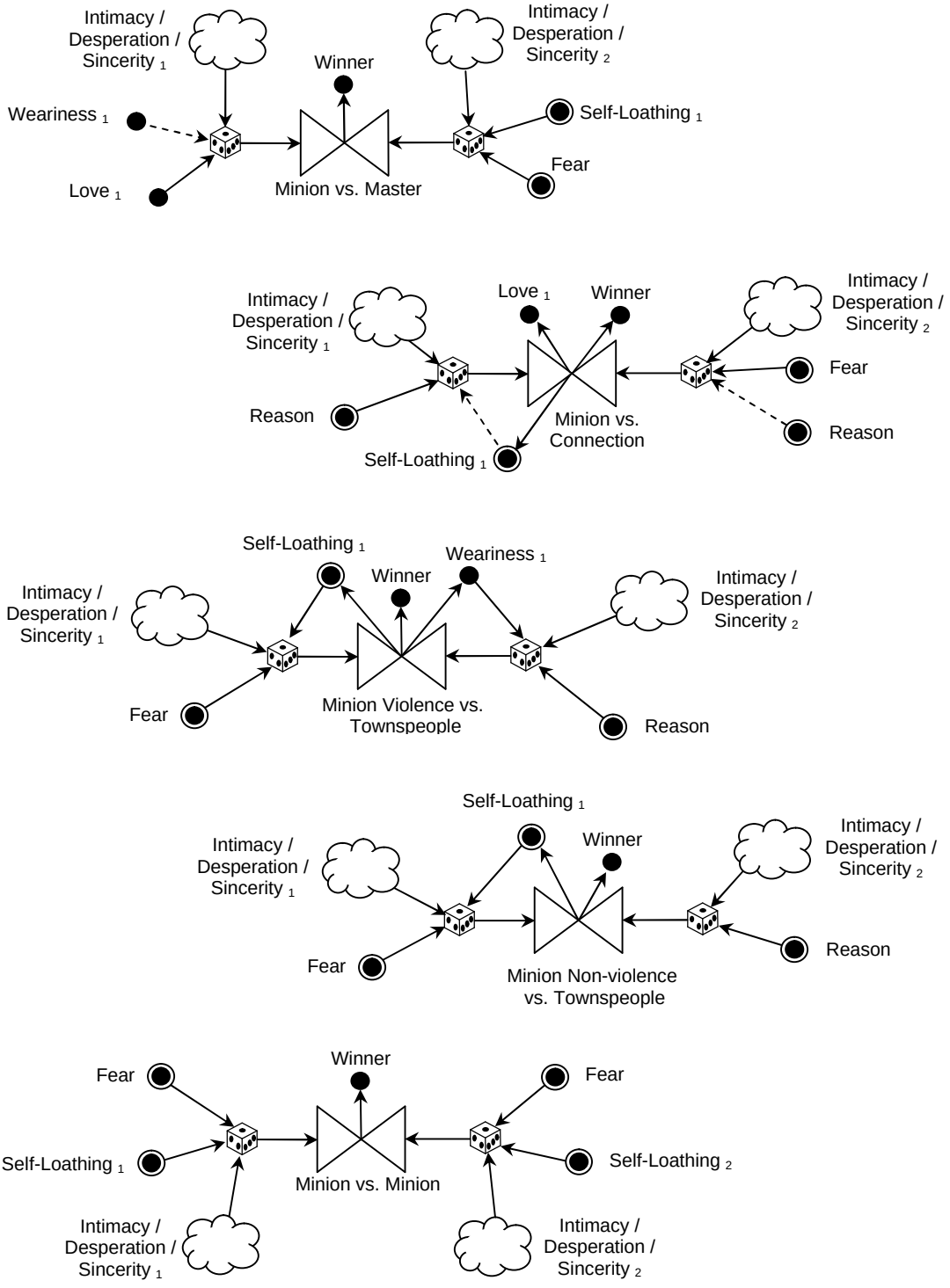


The player characters, or minions, of My Life with Master have three numeric attributes of Self-Loathing, Weariness, and Love. The evil master has the single attribute of Fear and the townspeople collectively have the attribute of Reason.

Minions also have More than Human and Less than Human traits. These are characteristics or abilities that indicate the situations in which the minion cannot fail or cannot succeed. For example, if a minion has a More than Human trait of "Has super strength except when he hasn't eaten for 24 hours," then any test of strength will automatically succeed unless he has fasted for a day. Similarly, if his Less than Human trait reads "Is clumsy unless pursuing food," any test of grace or agility will always fail unless he is searching for a meal.

Finally, minions have Connections, which are relationship traits the character has with individual townsfolk. The sum of the ranks in these traits gives the minion's overall Love value. As minions visit their connections, they gain Love which helps them to break their master's psychological hold over them.

Conflict & Reward System



Note: When More than Human or Less than Human traits apply, success or failure is automatic.[1]

Conflict System

The five attributes of Self-Loathing, Weariness, Love, Fear, and Reason are used in various combinations to resolve conflicts. For example, to resist the master's orders, the minion must roll Love minus Weariness against his master's roll of Fear plus Self-Loathing (this is actually the minion's Self-Loathing as the master only has a Fear attribute). On the other hand, to carry out some violent order of his master, the minion must roll Fear plus Self-Loathing against his opponent's roll of Reason plus Weariness. The various formulae are hard to remember (as least for me), but they do follow a sort of horror movie logic if you study them carefully.

To resolve the conflicts, dice pools are used. These are made up mostly of d4s whose number is determined by the formulae previously mentioned. But, an additional die may be added through role-play as a form of Narrative Reward. Only one additional die may be awarded by the Game Master on any conflict. It is earned by role-playing Intimacy, Desperation, or Sincerity. An extra d4 can be earned through intimacy, which essentially boils down to sitting down and having a glass of wine or meal with your opponent, giving gifts, or other similarly endearing actions. A d6 can be earned through desperation, which involves the character begging or pleading his case with extreme emotion. A d8 is earned through sincerity, the demonstration of real concern for the opponent. Concerning sincerity, the game text says, "you'll know it when you see it." The master is incapable of sincerity, so the players can always trump the master in this regard.

Once the dice pools are gathered, both sides roll the dice and sum the results. Excluding the narrative reward die, any rolls of 4 count as zero. The higher overall total wins the entire conflict and gets to narrate the outcome. So, there is only one contest roll per scene.

The fact that different formulas are used for different kinds of conflicts complicates the diagrams used to illustrate the conflict resolution system. Ordinarily, we could just stick a gauge labeled "Pertinent Attribute" into one or two diagrams to simplify things, because most games use their attributes in a similar fashion throughout. But, *My Life with Master* does not. So, we actually need five diagrams to illustrate the conflict resolution system: one for each type of conflict that can arise in the game.

Turn Order

Given that conflicts are completely resolved through a single roll, there are no rules for initiative or action order within a scene, nor do there need to be. There is a pre-determined ordering to the scenes, though (see the Structured Story design pattern), which dictates when players contribute and the general nature of the scenes.

Reward System

Whenever a minion makes an overture to a connection, he gains a point of Love, whether he succeeds in his contest roll or not. If he fails, he also gains a point of Self-

Loathing. Self-Loathing is also raised whenever the minion successfully enacts a violent or villainous act in service to his master. Since Self-Loathing is conflicted, with higher values making it harder for the minion to resist his master's orders, the raising of its value cannot truly be considered a reward. From the perspective of achieving the endgame, it is actually a punishment.

The only other rewards in the game are the narrative rewards given for role-playing Intimacy, Desperation, and Sincerity. These final rewards are perhaps the most important to game play, however, because they represent the only real control a player has over the success or failure of any given conflict.

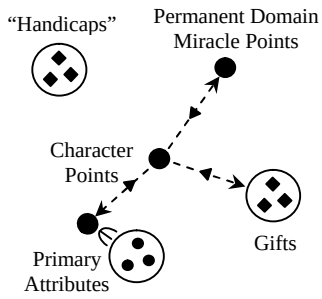
Nobilis

Nobilis was written by R. Sean Borgstrom and is published by Hogshead Publishing Ltd. It is a game in which players portray Nobilis, demi-gods in service to some great Power, or Imperator. The Imperators continually fight their archenemies, the Excrucians, godlike beings existing outside of Creation and bent on its destruction. This never-ending war rages mainly in the spirit realm, but occasionally spills into the mortal realm. Protecting Earth is the responsibility of the Nobilis. The game is rich with flavor and sparing in rules. The only time rules come into play is when the miraculous actions of the god-like characters (player and otherwise) conflict. And, even then, the system does its best to remain as non-intrusive as possible to enhance the mood of clashing divinities. Toward this end, the game entirely dispenses with dice and similar devices. After all, for gods a mere wave of the hand summons grand miracles. Why should players differ?

RPG Design Patterns Identified

Anonymous Rule (as Hit Points fall, Miracle Point costs rise and injuries are harder to avoid), Diceless, Faction ("Affiliation"), Flaw ("Handicap", "Restriction"), Drama Based Initiative, Game Master ("Hollyhock God"), Generalized Contest, Gift, Hit Points (in three flavors of "Surface Wound," "Serious Wound," and "Deadly Wound,"), Last Man Standing, Point Spend Attribute, Resource, Trauma Gauge (see Anonymous Rule Pattern).

Character Makeup



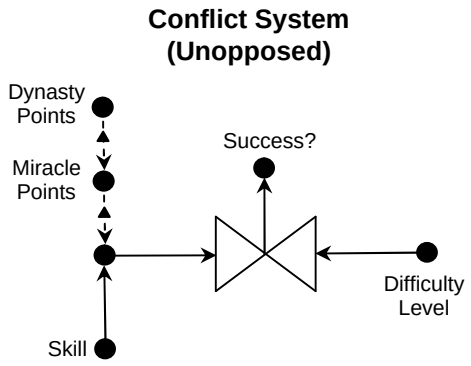
Characters have four primary attributes ranging in value from 0 to 5 and are set by spending character points from a pool. These are "Aspect," "Domain," "Realm," and "Spirit." Aspect controls physical, mental, and social prowess. Domain reflects the character's ability to manipulate his "Estate," which is the demi-god's domain of power such as "Forest" or "Light." "Realm" indicates how well the character governs his "Chancel," a demesne within the confines of Earth acting as a permanent place of residence. A high Realm indicates the character can mold and knead physical laws to his will within the Chancel. Finally, "Spirit" describes how well the character controls magical powers, including his miraculous defenses against all forms of attack. Characters are also assigned four pools of "Miracle Points" which are directly tied to the four attributes. Miracle Points are spent to obtain success in conflicts. Players may also spend their initial Character Points to raise their permanent "Domain Miracle Points" value, which determines the number of temporary Domain Miracle Points a character starts with each gaming session.

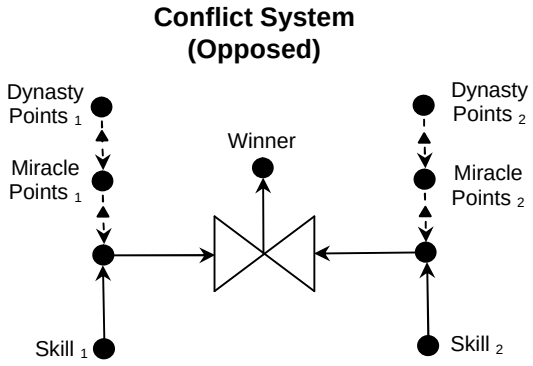
Characters also have Restrictions, Virtues, and Affiliations. Restrictions are essentially minor weaknesses such as "Cannot Kill" or "Honest" that sometimes hamper a character in his goals. Adopting a restriction gives a player more character points to

spend. Virtues are strengths with their own benefits, such as “Spiritual” or “Artisan.” The absolute nature of Virtues can land a character in trouble because a character cannot deny or contradict his Virtue. Affiliations, or “Codes,” are rules and beliefs by which a character lives.

Conflict System

Actions that do not deal with miracles are handled in a purely free form fashion. Only miraculous actions require resolution. All miracles are given a difficulty rating (“Level”) of 0 to 9 depending on the desired effect. Normally, characters perform miracles at a rating equal to their pertinent attribute at no cost. However, they can raise this value by spending Miracle Points from the pool associated with the attribute. When miracles oppose one another, the higher rated miracle always wins. However, a player can only spend 1, 2, 4, or 8 Miracle Points at a time. So, if a character has a Domain of 2 and his player wants to perform a miracle requiring a Domain of 5, he must actually spend 4 points rather than 3. What this means is that the greater the difference between the character’s attributes and the miracles he wants to perform, the more costly it becomes. These pools usually refresh at the end of every scene, but that is left to the discretion of the game master.





Turn Order

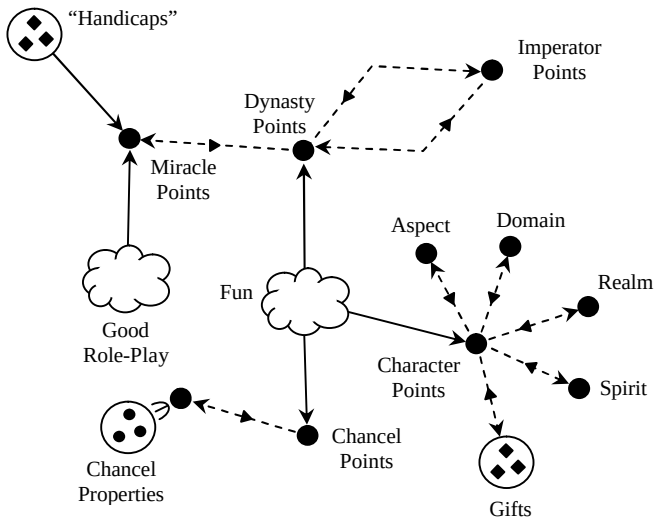
The order in which actions occur is entirely unconstrained in *Nobilis*. The game master decides who goes when based on what he feels is dramatically appropriate.

Reward System

One means of rewarding players is based on how much fun the players had. At the end of every game session in which all of the players have fun, the game master awards each player a “Dynasty Point.” At the end of every story in which all players had fun in most of the sessions, each character gains either a “Character Point” or a “Chancel Point.” “Fun” is measured simply by asking the players whether they had it. Character points can be spent to raise the four core attributes of Aspect, Domain, Realm, and Spirit or can be used to purchase gifts. Dynasty Points can be used in place of any Miracle Point when the character’s reserve runs dry. They are precious, though, in that they do not replenish as do Miracle Points. So, they are only used as a form of emergency backup. Chancel Points are pooled together by members of a Chancel to

buy “Chancel Properties.” This equates to roommates pooling their funds to buy a better television or stereo.

Reward System



More interesting rewards involve the earning or loss of Miracle Points based on a character’s “Handicaps,” which are comprised of “Limits,” “Affiliations,” “Restrictions,” and “Virtues,” A character earns Miracle Points whenever his Restrictions or Virtues hinder him in some significant way during play. A Limit is a restriction placed upon a character’s powers. Whenever Miracle Points are normally regenerated (such as at the beginning of a story), a character’s Limits will earn him more Miracle Points. When a character performs an action in accordance with his Affiliation’s “Code,” he earns a Miracle Point. When he defies it, he loses a point. (Affiliations are actually the *primary* means of obtaining Miracle Points in the game, and are a very strong incentive to role-play.) These rewards all have a potent impact on the game because they encourage players to seek out situations in which their Codes, Restrictions, and Virtues apply. That way, they can role-play their characters' weaknesses, idiosyncrasies, and beliefs while racking up the Miracle Points.

Paranoia xp

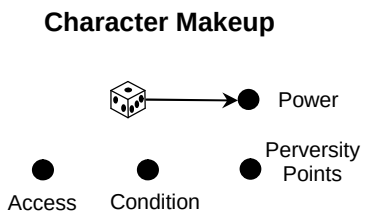
Paranoia xp was designed by Allen Varney and is published by Mongoose Publishing. It is derived from the original *Paranoia* game, which was first published by West End Games in 1984 and was designed by Dan Gelber, Greg Costikyan, and Eric Goldberg. It is a comical game set in the "Alpha Complex" in a nightmarish bureaucratic future where everyone truly is "out to get you." Alpha Complex is dispassionately run by The Computer whose primary function is to ensure the happiness and well-being of everyone under its control. Unfortunately, The Computer is a few diodes short of a radio. It believes there are traitors, mutant traitors, and Commie bastard traitors living amongst the loyal citizens of Alpha Complex that must be dealt with. That is, killed. As a newly appointed member of the elite Troubleshooters, it is your job to root out and destroy all traitors. The trouble is, you are one yourself. In *Paranoia xp*, Death is overworked and cloning technology is state-of-the-art. Characters drop like peanut shells in a circus tent, but replacement clones appear within minutes.

RPG Design Patterns Identified

Attribute, Game Master, Generalized Contest, Last Man Standing, Rank, Recycled Fortune ("Conflict Roll," "Tension Roll"), Resource, Skill, Structured Session, Trauma Gauge

Character Makeup

All new characters are Troubleshooters. Troubleshooters are granted "Red" clearance. This is much better than the "Infrared" clearances possessed in their former meaningless pre-game lives and is an indication of how much The Computer trusts the characters in their new meaningless in-game lives.



Questions about character design are of the highest classification level: "Ultraviolet." Attempting to access information above your classification level is treasonous. Only the Game Master and High Programmers are classification level Ultraviolet. If you are of a lower classification level and are reading this description of *Paranoia xp*'s Character Makeup, you are obviously a traitor. Kindly proceed to the armory and requisition an XJ7-5 repeating blaster pistol by filling out form G237-9Q/B in triplicate. Once the XJ7-5 repeating blaster pistol is obtained, place the muzzle to your temple and pull the trigger. Failure to comply proves your treasonous nature and is grounds for summary execution. Note: For your protection and well-being, the XJ7-5 repeating blaster pistol possesses a safety mechanism covering the trigger to prevent accidental discharge. Only citizens possessing security level "Green" and above may remove this mechanism, demonstrating how much The Computer values your safety and health. You should always trust The Benevolent Computer.

Since you have read this far, you clearly must have Ultraviolet clearance. Very well. *Paranoia xp* character stats are generated by the Game Master and are never revealed to

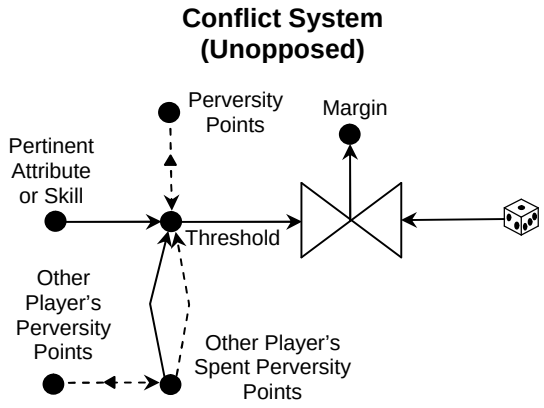
players. Characters all have two primary attributes. These are “Access” and “Power.” Power is determined by rolling 1d20. If a value less than 8 is rolled, Power is set to 8. Power is a gauge of the character’s skill in using his mutant power. Yes, all characters are mutants. Yes, mutants are traitors. Yes, it is the duty of all characters to kill traitors, but *only* if they have sufficient proof of the traitor’s treasonous activities. The Access attribute always starts at 1. It is a gauge of how well the character can “work the system” to get what he needs or wants. It also acts as a form of “Treason Armor” that reduces the harsh punishments meted out for the many treasonous actions the character will undoubtedly perform.

Characters have a resource called Perversity Points. Perversity Points can be spent to improve the odds of success in conflicts. The points are the primary reward of the game because, aside from their money, they are the only points players (supposedly) know about.

Finally, characters have a “Condition” attribute which takes on the values of “Okay,” “Snafued,” “Wounded,” “Maimed,” “Down,” “Killed,” or “Vaporized.” Each of these states corresponds to various detrimental effects which generally involve the inability to spend Perversity Points for a time. The character’s condition thus follows the Trauma Gauge Design Pattern. However, it does not act as a form of hit points because various injuries are not cumulative. If a character sustains two lacerations rated at the Wounded level, the character is still just Wounded. In addition, a character can be Maimed or even Down without having first been Snafued. The only real difference between Killed and Vaporized is the entertainment value.

Conflict System

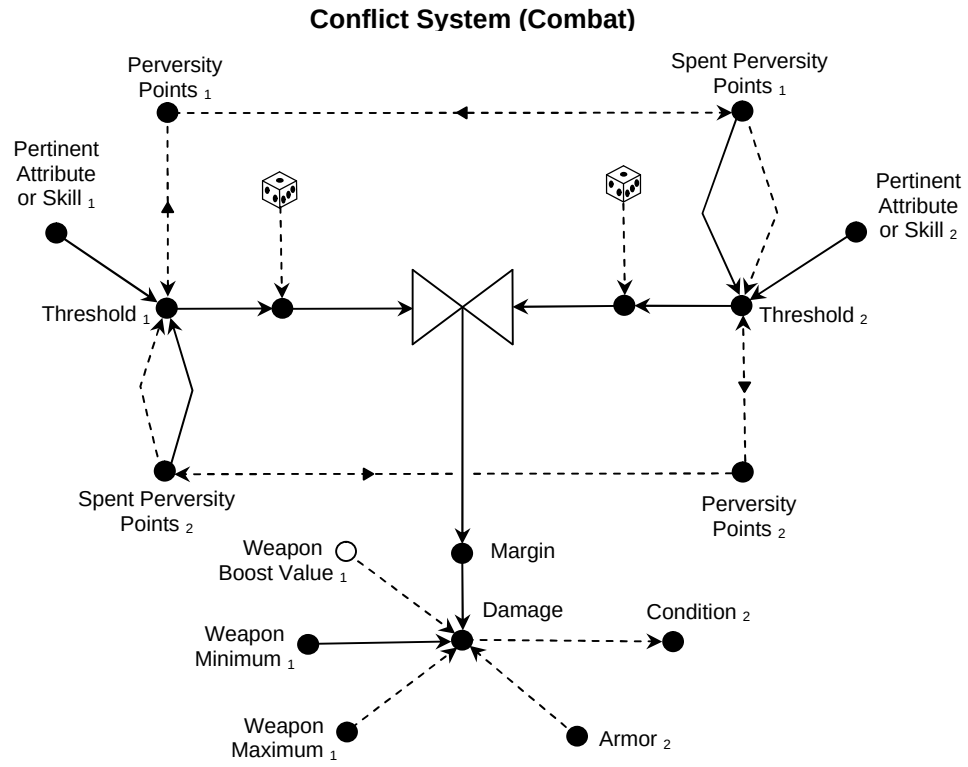
Attribute and skill rolls are performed by rolling a d20 and comparing it to a threshold. If the roll results in a value less than or equal to the threshold, the roll succeeds. Otherwise it fails. The difference between the roll and the threshold is called the “margin.” The margin determines how well the character succeeded or how badly he bungled his action. By default, the threshold is set to equal the attribute value or skill rank pertinent to the action. However, players can modify the threshold or the thresholds of others by spending Perversity Points, if the GM allows it. Modifications can be both positive and negative.



If one character pits his attributes or skills against another, an opposed roll results. Each side of the contest rolls a d20 against the pertinent attribute value or skill rank used. The side with the higher successful margin wins. If neither side succeeds or the

margins are equal, nobody wins. The contest can continue with another roll if the GM deems it so.

Physical attacks of all forms use the "Violence" skill. If such an attack is successful, the target sustains damage according to the type of weapon used. Each weapon has a minimum and a maximum damage value. It also has a "Boost" value. The margin of the attack is divided by the Boost value and rounded down. The result is added as a damage bonus to the weapon's minimum damage. If this value exceeds the weapon's maximum damage, the result is set at the maximum damage value instead. Armor can reduce this value.



Contests involving treasonous accusations work in exactly the same way as physical attacks, except the "Management" skill is used in place of the Violence skill. Different kinds of accusations inflict different amounts of "Treason Damage," which results in various "condition" levels similar in effect to those for physical wounds: "Okay," "Probation," "Censure," "Medication," "Brainscrub," "Termination," and "Erasure."

As an optional rule, a GM may decide to drain points from character attributes when they succeed in using those attributes. So, if a character manages to use his mutant power, he may lose Power points. To thumb its nose at "justice," the game suggests that low margins, which result in slight successes, should cost more than spectacular victories. That way, if a character is on a roll, he will continue in that fashion. But, if he achieves only minor wins he will spiral down until he continually fails at everything he attempts. To quote the game text: "…That's life in Alpha Complex…" Characters

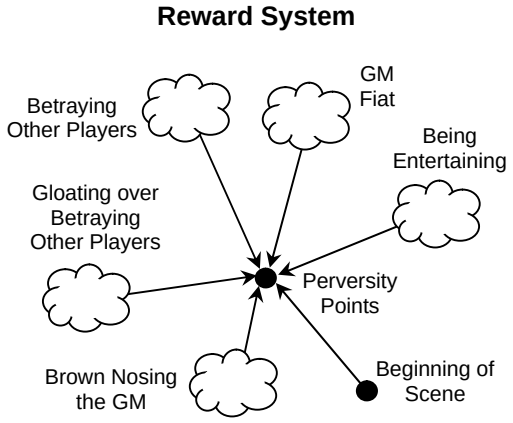
recover lost attribute points over time. If this optional rule is used, Power and Access act as Resource Attributes.

Each scene has a "Tension Level," which determines the odds of someone witnessing a treasonous act. Higher clearance level areas naturally warrant more cameras and more sophisticated monitoring equipment, so The Computer is more likely to observe characters in those areas. Due to this, the location in which a scene occurs determines the tension level, which ranges in value from zero (0) to 20. The higher the number, the more tension exists. So, supply closets, garages, and Red clearance areas warrant low values. Ultraviolet and Computer Core areas demand high values. True to form, the most extensive surveillance equipment is positioned in Alpha Complex's bathrooms. When a character performs a treasonous action, he makes a contest roll as normal. After the conflict is resolved, the GM compares the d20 roll (without rolling again) to the scene's Tension Level. If the d20 roll is less than or equal to the Tension Level, the act was witnessed. So, brilliantly successful treasonous acts are almost always observed.

Turn Order

In contests involving more than two characters, all players make contest rolls as normal. The margins determine the degree to which each succeeds or fails in the contest. If the actions must follow a sequence, events are ordered by margin values with the highest margin going first.

Reward System



The GM awards a "ration" of "Perversity Points" at the beginning of every new scene. They are also handed out to players for being entertaining, for betraying other player characters, for gloating over the betrayal of other player characters, for brown nosing the GM, and for whatever the heck the GM wants to dole them out for. The game text makes the point that the rewards are given to the player, not to his character. So, when a character dies (not *if*, but *when*) the player retains the points he has accumulated.

The Pool

The Pool was written by James V. West. As of this writing, the game is freely available for download at http://www.randomordercreations.com/thepool.html. It is a game that is a model of brevity. In fact, the entire game text is not much longer than this description of it. The game has no pre-defined setting and is focused on creating interesting, well narrated stories.

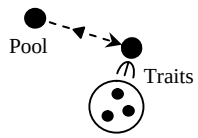
RPG Design Patterns Identified

Dice Pool, Drama Based Initiative, Game Master, Negotiated Contest, Rank, Resource, Shared Power, Trait

Character Makeup

Character generation starts by the player writing up a 50 word story about his character. Fifty words is not much to work with, so the player must focus on his character's most important aspects. These elements are translated into traits. These traits can be assigned ranks by sacrificing dice out of the character's Pool, which starts out containing 15 dice. Just look at that gauge diagram for Character Makeup. Can it get much simpler?

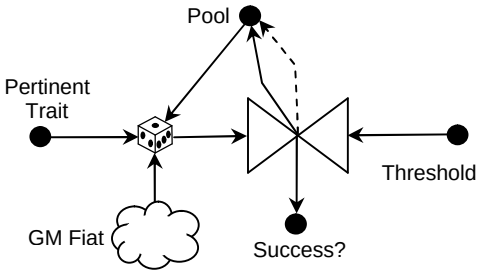
Character Makeup



Conflict System

Conflicts rolls can be called by any player, not just the game master. Conflicts are resolved by rolling dice pools of d6. The number of dice rolled equals the rank of the character in any single applicable trait plus 1 to 3 d6 provided by the game master plus any number of dice (up to 9) from the character's Pool. Any roll of 1 on any die indicates the player earns a "monologue of victory." The player essentially earns the right to describe the outcome of the conflict. This does not equate to "success," though, because the player is perfectly free to describe how his character failed. If no 1s are rolled, though, all dice used in the conflict that were drawn from the character's Pool are permanently lost. Thus, it is always a gamble for a player to draw from his character's Pool. The result of any conflict roll covers the broad consequences concerning the player's intent. That is, it covers much more than the single swing of a sword. It can cover the success of an entire battle (although the game is not focused specifically on combat).

Conflict & Reward System



Turn Order

The game foregoes any explicit rules for determining when and how often players perform actions. The game master assumes that responsibility, although the written

rules only infer this point. In responding to a request for clarification, the game's author, James V. West, elucidates: "If a player says he wants a roll, let him." By observing the other players' reactions to the resulting monologue, the game master assesses if he "should stop him at some point or let him go the whole nine yards." In other words, the game master decides whether any given roll applies to a brief sequence of actions or the slaying of the entire dragon. According to Mr. West, if circumstances arise where multiple players interact within a scene, the game master should simply wing it. Like the game *Nobilis*, *The Pool* relies entirely on dramatic impact to decide the ordering of player actions.

Reward System

Whenever a player succeeds on a roll, he must choose between narrating the outcome of the conflict or adding a single die to his character's Pool. Pool dice can be sacrificed to raise the character's rank in his various traits.

Puppetland

John Tynes authored *Puppetland* and Hogshead Publishing distributes it. In this game, all of the players portray puppets in a world fashioned by the Puppet Maker. Unfortunately, the puppet Punch murdered the Puppet Maker and thus because the primary villain of the game. Punch has a number of minions, known as "Punch's Boys," which he fashioned from the flesh of the dead Puppet Maker. Punch also has an army of nutcracker soldiers. He tyrannically rules most of Puppetland without compassion or mercy. The game encourages the players to overthrow Punch and, if possible, restore the kindly Puppet Maker to life.

The world contains a number of fairy tale attractions. First, most puppets live in Puppettown, which Punch's castle overlooks. The Lake of Milk and Cookies borders Puppettown. True to its name, this lake contains eternally yummy milk in which cookies float like icebergs. Across the lake from Puppettown lies Respite, a small village founded by Judy, a kindly female marionette who was Punch's one time lover but is now his arch enemy. Puppets fleeing Punch's wrath congregate here.

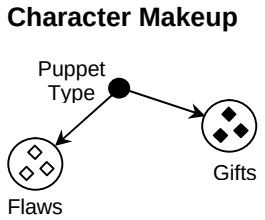
The game world follows the rules of puppet physics. That is, events take place purely for dramatic purpose rather than as any kind of attempt to simulate "reality." There are no tables, no dice rolls, and no numbers to compare. If one puppet stabs another puppet with a sword, the second puppet falls to the ground. At the end of each hour-long session, all puppets in Puppetland wake up safe and snug in their beds fully mended, even those recently stabbed or otherwise mutilated. The only exception to this rule is that puppets that have lost all of their "puzzle pieces" do not waken. Puppets start out with 16 puzzle pieces and lose one whenever injured or if they attempt an action their description says they cannot do (such as pick something up when they have no hands). Once lost, puzzle pieces never return.

RPG Design Patterns Identified

Class (various puppet types), Diceless, Drama Based Initiative, Faction (Punch's Boys vs. everyone else), Flaws, Game Master, Gifts, Hit Points ("puzzle pieces"), Negotiated Contest, Traits

Character Makeup

Characters can be Finger Puppets, Hand Puppets, Shadow Puppets, or Marionette Puppets. The game recommends having at least one of each type in a group. The various puppet types have different strengths and weaknesses. Finger puppets lack hands, so they cannot grab things. However, their tiny stature makes them quick, so they can dodge aside when things are thrown at them. Shadow puppets disappear when they stand sideways due to their extreme thinness. The big, strong, and slow marionettes can pick things up and throw them, but cannot dodge. So, each of these puppet types



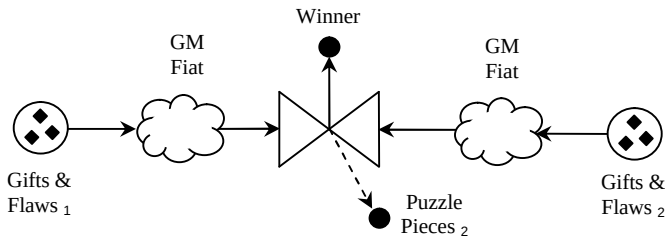
has specific flaws and gifts categorized under the headings of “This puppet is,” “This puppet can,” and “This puppet can not.”

After choosing the puppet type, the player must decide on three traits about the puppet for each of the gift and flaw categories. These can be anything the game master agrees to and are commonly such things as “very clever,” “do magic tricks,” “can’t tell a lie,” etc.

Conflict System

The decision of who wins a conflict is entirely on the game master’s shoulders. He makes choices based on what he feels is most dramatically appropriate to the story. A character’s flaws and gifts ordinarily weigh heavily into these judgments.

Conflict System



Turn Order

Action order is, once again, based purely on the dramatic effect it has on the storyline and is decided upon by the game master.

Reward System

Lacking virtually all mechanics, *Puppetland* essentially has no mechanical reward system. Its hit point system, involving the puppets’ puzzle pieces, is about the closest thing that it has. Puppets don’t want to take damage, since it is permanent and always brings a puppet closer to his “death.” So, avoiding damage is rewarded by virtue of the fact that the puppet didn’t take any.

The Riddle of Steel

The Riddle of Steel was written by Jacob Norwood and Rick McCann and is published by Driftwood Publishing. It is a dark and gritty fantasy game set in the world of Weyrth where combat is swift and lethal. The game claims to have the most realistic RPG combat system ever devised. Some gamers would argue that point, of course, but the game can lay claim to the fact that ARMA (The Association for Renaissance Martial Arts) has given the game its stamp of approval. Of course, "realistic" does not always equate to "fun," but the game has a growing audience and many players certainly find it appealing.

RPG Design Patterns Identified

Currency, Game Master, Generalized Contest, Hit Points ("Health"), Idiom ("Spiritual Attributes"), Last Man Standing, Point Spend Attributes, Priority Grid (character generation), Rank, Skill, Trait ("Spiritual Attributes"), Wound Trait

Character Makeup

In creating a character, a player must decide upon the order of his priorities for the character. The various priorities are: "Race and Sorcery," "Social Class," "Attributes," "Skills," "Proficiencies (& Vagaries)," and "Gifts and Flaws." The player essentially ranks these in any order. Once these are set, the player knows what choices he has in each priority. For example, the only way for a player to obtain a magic-wielding Fey character is to set "Race & Sorcery" as the highest priority. This, in turn, means the character sacrifices in other areas. He may have fewer points to spend in Attributes, have a lower social standing than other characters, or have fewer gifts or more flaws.

Characters have five "Temporal", five "Mental", and five "Derived" attributes. During character generation, players have a number of points to distribute between the Temporal and Mental attributes. The quantity of points he has to distribute depends on how high a priority he sets on the character's Attributes versus other concerns. The Temporal attributes are: "Strength" (ST), "Agility" (AG), "Toughness" (TO), "Endurance" (EN), and "Health" (HT). The Mental attributes are: "Will Power" (WP), "Wit" (Wit), "Mental Aptitude" (MA), "Social" (Soc), and "Perception" (Per). The "Derived" attributes are: "Reflex," "Aim," "Knockdown," "Knockout," and "Move." The values of the derived attributes are calculated from the Temporal and Mental attributes by various formulae. In addition, characters have a number of other attributes, including "Insight Points," "Fatigue," "Pain," and "Bloodloss."

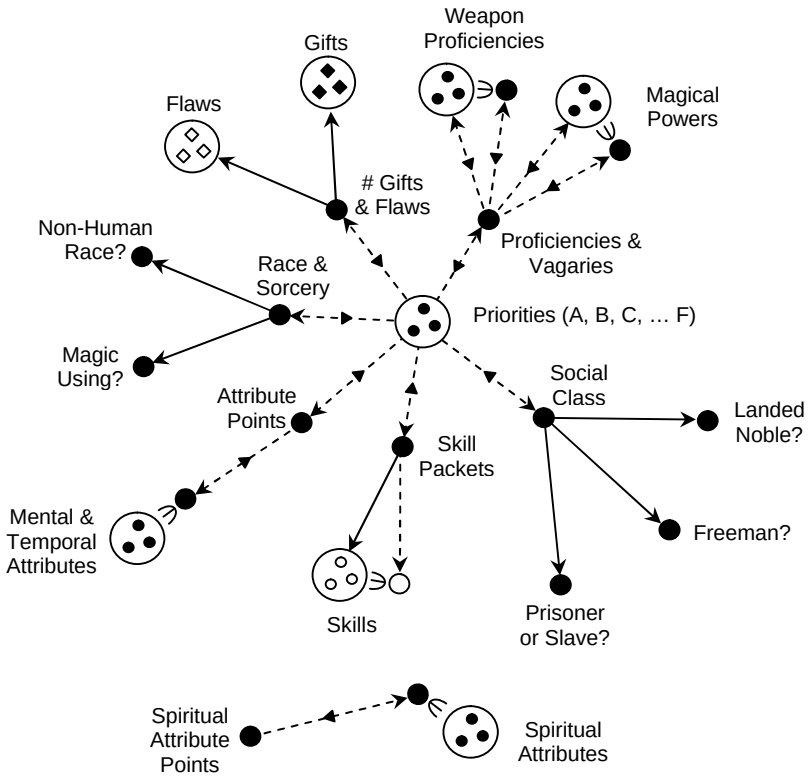
Most importantly, characters possess five "Spiritual Attributes" taken from a list of 6 possibilities, including: "Conscience," "Destiny," "Drive," "Faith," "Luck," and "Passion." Some of these options can be chosen more than once. Players have seven points initially to distribute among those selected. Since they are selected by the player rather than being universal for all characters, Spiritual Attributes are not technically attributes according to the definition given in this book. Rather, they are traits. The Spiritual Attribute categories themselves are well defined, so they almost satisfy the definition of "skills" rather than "traits." However, once they are selected, the actual

traits are customized by the player according to his character concept. These spiritual traits provide guidance on how the character should be role-played and form an important basis for rewards in the game. So, Spiritual Attributes would actually be classified as Ranked Idiom Resource Traits. That's quite a mouthful, but they bring together a lot of concepts and truly form the heart and soul of *The Riddle of Steel*, so it is not surprising that they pack a powerful punch in design pattern terms.

The Insight Points attribute equals the number of points that have been spent from the Spiritual Attributes on various other skills and attributes. Insight allows bonuses on the next character to be created after the current character dies or retires.

Finally, characters have lists of skills, gifts, and flaws.

Character Makeup



Conflict System

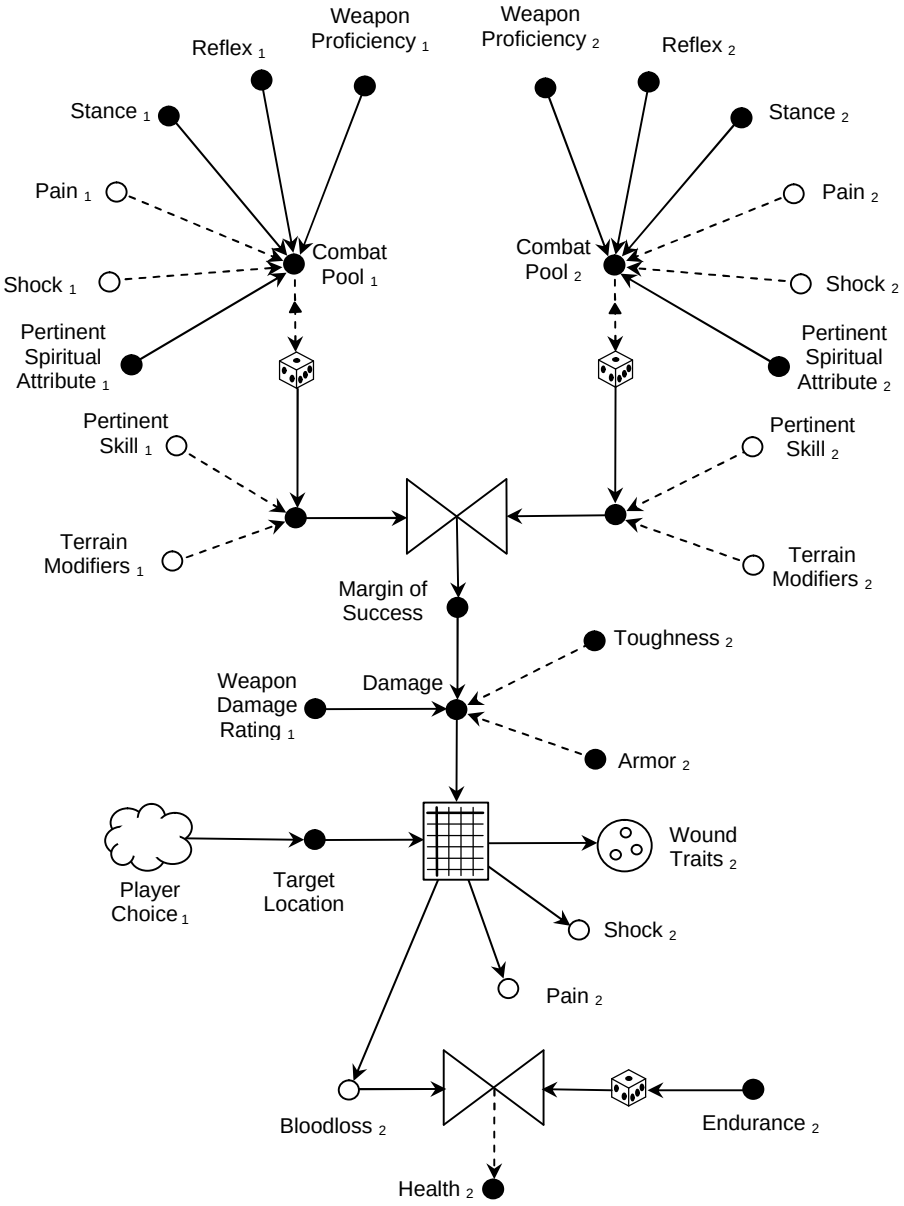
Conflicts are resolved using dice pools of d10s, whose size incorporates various factors such as Reflex, Weapon Proficiencies, the all-important Spiritual Attributes, and other considerations. The dice are rolled and the values noted. Any rolls of 10 indicate that die should be rolled again and the results accumulated ("Stacked") for that die in an open-ended fashion. The resulting values are then compared against a threshold ("Target Number") indicating the difficulty of the action. Any values that equal or exceed the target number count as successes. For skills, the target number equals the rank of the skill, so low numeric ranks in *The Riddle of Steel* indicate superior capability. Generally, only one success is needed, but multiple successes serve as a gauge of the magnitude of victory. If a roll fails and the dice come up with two or more 1s, the character fumbles or botches the action.

Uncontested actions go no further. However, if there are two active participants competing against one another, an opposed roll must be made. The participant with the greatest number of successes wins. The degree to which he wins is determined by the difference in the number of successes between the two participants. This difference is called the "Margin of Success." So, if one character obtains 3 successes on his roll and another obtains only 1 success, the first character wins with a Margin of Success of 2.

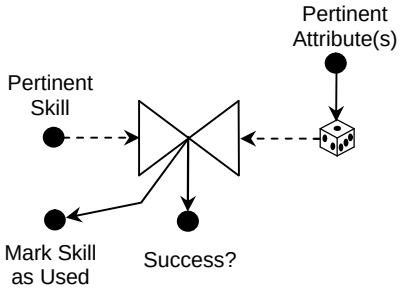
Extended contests take place over a period of time and require a number of contest rolls. These require a character to accumulate a certain number of successes before the task completes. Any fumble forces the character to start over.

Fighting boils down to an extended contested conflict between two combatants. On a successful blow, the hit location is determined and then damage is calculated by adding the Margin of Success to the weapon's damage rating. This is adjusted by the character's Toughness and Armor. The result is then used in a table lookup to determine the actual wounds inflicted. The table to be used in this lookup depends on the hit location. Damage comes in the form of actual wound descriptions (i.e., "Serious Flesh Wound"), and Shock, Pain, and Bloodloss points. Shock is subtracted from the target's dice pool when the wound is inflicted but has no lasting effect. Pain is subtracted from the target's dice pool at the beginning of every combat round. Also, at the beginning of every round, any wounded combatants must make a roll of Endurance against Bloodloss. Failure indicates the character loses one point of Health. When his Health drops to zero, the character dies.

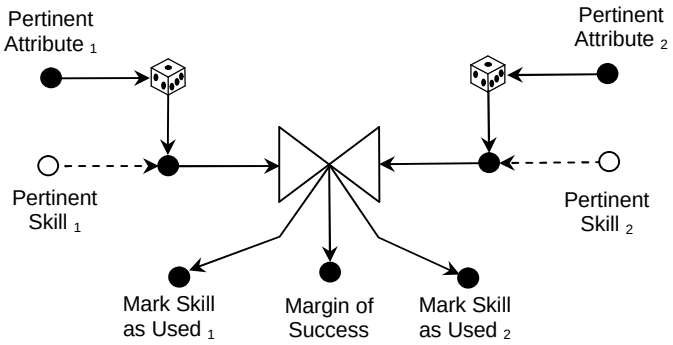
Conflict System
(Combat)



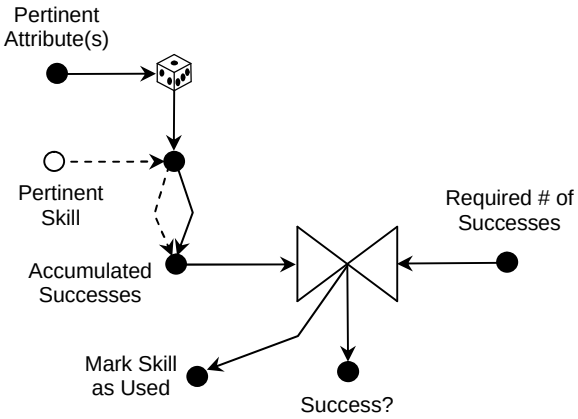
Conflict System (Simple Unopposed)



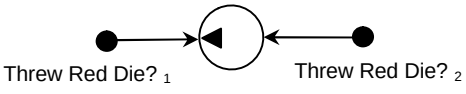
Conflict System (Simple Opposed)



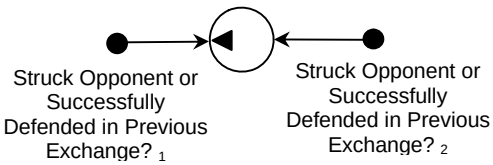
Conflict System (Extended Unopposed)



Turn Order (first exchange)



Turn Order (subsequent exchanges)



Turn Order

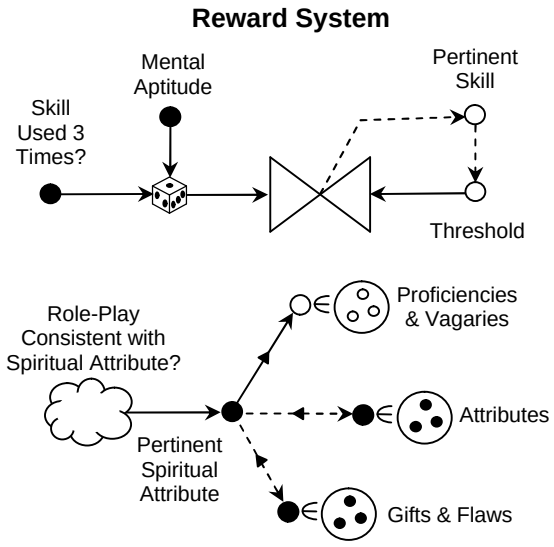
Every round of combat is divided into two "exchanges," each of which is basically the amount of time needed to make an attack. At the beginning of a fight, each combatant declares the "stance" that his character assumes. Various stances give differing bonuses to offensive and defensive maneuvers. The stances only last throughout the first exchange.

Initiative is initially determined by having each player conceal a colored die in his hand. Red means he is going to attack, white means he is going to defend. When the Game Master ("Seneschal") shouts "throw," each player throws out his colored die. Any hesitation on a player's part means his character can only defend. If neither attacks, the characters are circling one another looking for openings. If both players attack simultaneously, neither presents adequate defenses against their opponent. Such situations are unhealthy for both parties. Characters are allowed to taunt their opponents to force them to attack after a sufficient amount of circling has transpired.

Once melee is engaged, the aggressor maintains the initiative as long as he keeps striking his opponent. Only the aggressor can attack. The defender uses his abilities exclusively to avoid being hit. The first time the aggressor misses, however, the roles reverse and the defender becomes the aggressor.

Reward System

Every time a character uses a skill (falling under the category of "Skill") in a dangerous or stressful situation, his player puts a check mark next to the skill. When the character accumulates three check marks, the player makes a conflict roll of Mental Aptitude versus a threshold of 15 minus the skill rank. If he succeeds, the skill's rank lowers by one (remember, lower ranks indicate superior ability in this game).



Rewards are also given through a character's Spiritual Attributes. If a player has his character act in a way that is consistent with his Spiritual Attributes, he is rewarded by one or more ranks being added to the pertinent trait. For example, a point of Conscience is added whenever the character "does the right thing." Points can also be lost if the player portrays his character in a way that contradicts his Spiritual Attributes. Skills falling under the heading of "Proficiencies & Vagaries," attributes, gifts, and flaws can be bought (or bought down) by spending Spiritual Attribute points. Skills falling under the category of "Skills" are improved as described above.

RIFTS

RIFTS was written by Kevin Siembieda and is published by Palladium Books, Inc. It is a game set in a post apocalyptic Earth after nuclear holocaust wiped out billions of lives. During the conflict, so much psychic energy was supposedly released that the fabric of space actually tore along the forgotten ley lines of magic. These inter-dimensional gaps are known as Rifts. They provide the occasional nightmarish horror passage into our realm. Earth has become a world of potent magic, fantastic monsters, psychic powers, and high technology.

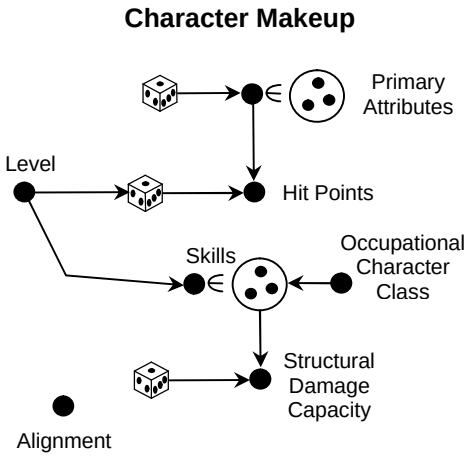
RPG Design Patterns Identified

Alignment, Class, Game Master, Generalized Contest, Hit Points, Last Man Standing, Level, Race, Random Attribute, Skill

Character Makeup

RIFTS characters have 8 primary attributes of "Intelligence Quotient," "Mental Endurance," "Mental Affinity," "Physical Strength," "Physical Prowess," "Physical Endurance," "Physical Beauty," and "Speed." Each attribute is generated by rolling 3d6 and summing the values. If the result is a 16, 17, or 18, another d6 is rolled and the result is added to the total.

All items have a hit points-ish resource known as "Structural Damage Capacity" (SDC) indicating how much damage is needed to break it. The SDC of a living creature must be taken down to zero before its "Hit Points" are affected. "Hit Points" fits the definition of the term used in this book. When hit points drop to zero, the character begins dying.



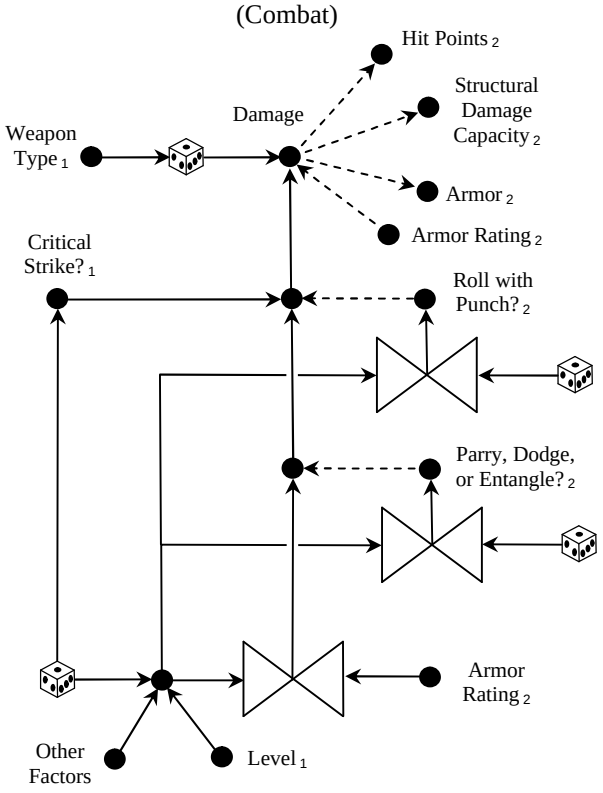
The game has a list of 125 skills partitioned into the various categories of "Communications," "Domestic," "Electrical," "Espionage," "Mechanical," "Medical," "Military," "Physical," "Pilot," "Pilot Related," "Rogue," "Science," "Technical," "Weapon Proficiencies," and "Wilderness." *RIFTS* also has 22 "Occupational Character Classes," including "Borgs," "Headhunters," "Vagabonds," "Techno-Wizards," and the like. Each class has attribute requirements that characters must satisfy to select it. Once taken, the class bestows certain skills on the character. Classes also provide lists of additional skills from which the player chooses for his character as he gains levels.

Players must choose from among three alignment categories of "Good," "Selfish," and "Evil" for his character. In each category there are sub-types. From the Good category a player may choose "Principled" or "Scrupulous." From the Selfish category, he may

select “Unprincipled,” or “Anarchist.” From the Evil category, he has the options of “Miscreant,” “Aberrant,” and “Diabolic.” The alignment system has no effect on the game other than to serve as a general guide on what kinds of actions are appropriate when role-playing a character.

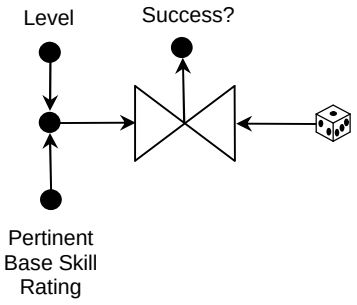
Conflict System

Conflict System
(Combat)



RIFTS resolves all combat conflicts using d20s. If a player attempts a special action, such as a “Death Blow” or “Knockout/Stun,” he must declare his intention before rolling. Rolls are adjusted by various factors. These include “Aimed Shots,” “Automatic Weapon Bursts,” “Shooting Wild,” whether the combatant has superior training in hand-to-hand combat, his level, etc. If the adjusted roll is 4 or less, the attack misses. Any value between 5 and the target’s Armor Rating (AR) indicates the attack hits, but only does damage to the opponent’s armor. Any adjusted roll greater than the Armor Rating indicates the attack delivers damage directly to the target’s SDC and hit points. If a hit is successful, the target has a choice of attempting to parry, dodge, or entangle the attack (Roll another d20 and beat the attack roll). If the defense is successful, the attack misses.

Conflict System
(Skill Use)

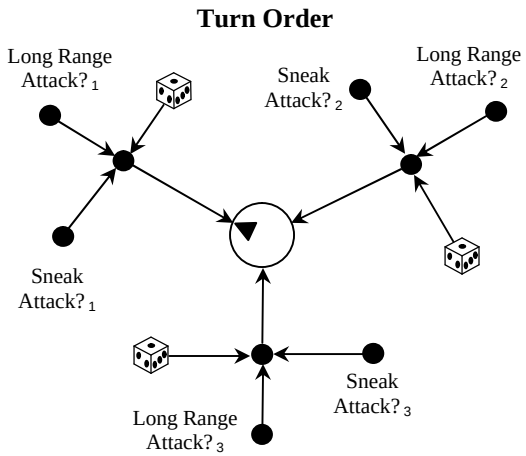


If an attack successfully lands, it delivers damage according to the type of weapon used. For example, a “Human Fist” delivers 1d4 damage while a “Bull Whip” inflicts 1d8 points of damage. High-tech weapons deliver “Mega-Capacity Damage.” The game equates each point of Mega-Capacity Damage to 100 points of normal damage. A Short Range Missile with a Light High-Explosive Warhead delivers 1d4x10 Mega-Damage. In other words, it delivers 1000 to 4000 damage. If damage is delivered via a blunt attack, the target can attempt to “Roll With The Punch” (Roll yet another d20 and beat the attack roll). If successful, this has the effect of reducing the amount of damage inflicted on the target.

Conflicts involving “Skills” are handled differently, with each skill specifying the formula to use in determining the character’s chances in using the skill. This formula is listed as a “Base Skill” rating plus 5% per level. For example, the “Disguise” skill has a chance of “25% + 5% per level of experience.” The “Base Skill” is an apparently arbitrary base percentage, presumably based on the inherent difficulty of performing the skill. So, proficiency in skills cannot be raised as individual ranks, but they do increase in effectiveness as a character gains experience levels.

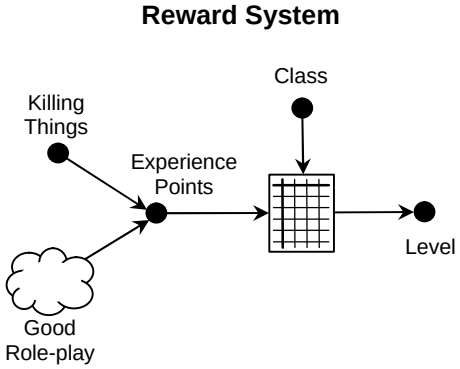
Turn Order

Characters performing “Sneak Attacks” or “Long Range Attacks” always go first. Otherwise, initiative is determined by having both sides in a conflict roll d20s. The side with the higher result goes first. Ties are re-rolled. The winning side goes and then the losing side goes. If either or both sides have multiple attacks every round, attacks are exchanged one at a time back and forth until all attacks are complete on both sides.



Reward System

The game rewards characters with experience points for overcoming foes and “good role-play.” A character’s level is based off of the number of experience points he has accumulated. Levels are gained at different rates based on experience point tables provided for each character class.



Rolemaster Fantasy Role Playing (Second Edition)

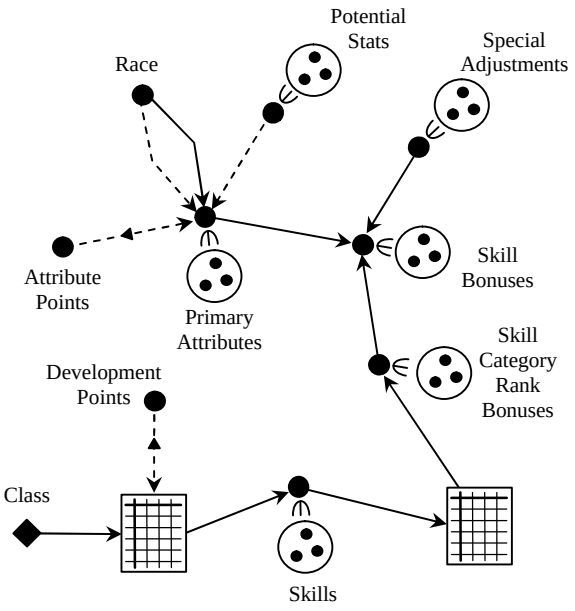
Rolemaster Fantasy Role Playing is published by Iron Crown Enterprises, Inc. Rolemaster is considered by many to be the poster child for highly detailed table driven combat systems. It is a combat-oriented, Tolkienesque fantasy game with men, elves, dwarfs, and halflings. Much of the supplemental material of the first edition was heavily focused on Tolkien's Middle Earth. (Actually, the Middle Earth supplements were written for a "Rolemaster-lite" game called *MERP – Middle Earth Role Playing* – published by the same company. But, the games were so similar that many people used the *MERP* supplements for Rolemaster games.) Unfortunately, the Lord of the Rings material was eventually dropped. (Rumor has it that this move was primarily due to prohibitive royalties demanded by the Tolkien estate for use of the content, but the author has seen nothing official verifying this supposition.)

RPG Design Patterns Identified

Alignment, Class, Game Master, Hit Points, Last Man Standing, Level, Point Spend Attribute, Generalized Contest, Race, Random Attribute, Rank, Success Reward (Experience Points)

Character Makeup

Character Makeup

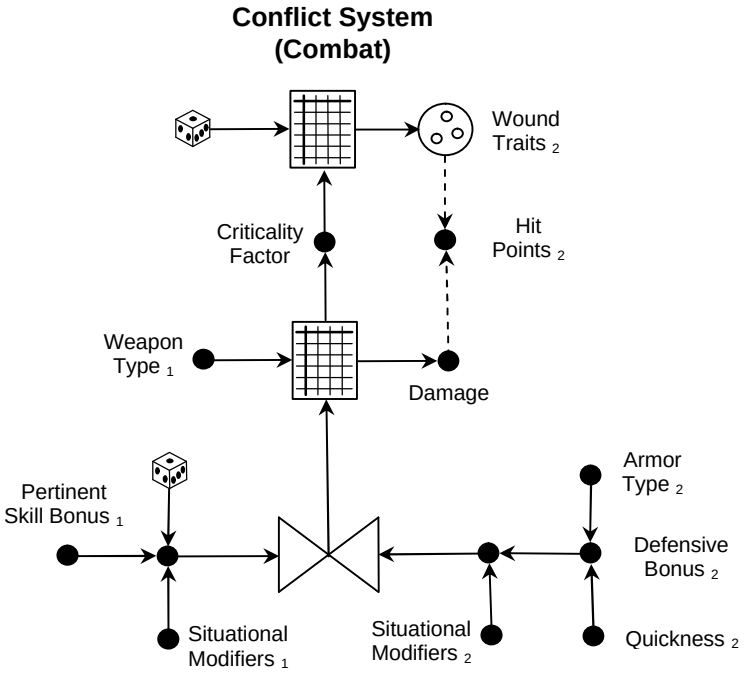


The power of a Rolemaster character is gauged as a numeric "Level," which rises as characters gain experience points. In addition, characters have 10 attributes ("Stats"): "Agility" (Ag), "Constitution" (Co), "Memory" (Me), "Reasoning" (Re), "Self Discipline" (SD), "Empathy" (Em), "Intuition" (In), "Presence" (Pr), "Quickness" (Qu), and "Strength" (St). These range in value from 1 to 101. Players initially set these values by spending points from a resource pool. (The size of the resource pool can be determined in various ways, but it ends up being around 660 points.) The amount to which these can be raised is limited by the character's "Potential Stats," which have no other function in the game than to indicate how good a character's attributes can possibly become. The attribute values are modified slightly depending on the character's Race ("Common Man," "High Man," "Wood Elf," "Halfling," etc.).

Characters also have a litany of skills. Each of these has a rank, which is purchased through the expenditure of "Development Points," or "DPs." The cost of each rank is based on the character's class ("Profession"), such as "Fighter," "Rogue," "Cleric," "Magician," etc. As character's gain levels, they earn more DPs that they can spend on raising the ranks of their various skills. Depending on the skill and class, some skills allow a character to only gain a single rank per level. In others, two or even three ranks can be purchased. Quite often, the cost to gain a second rank in a particular skill is considerably more expensive than the first. Again, these costs depend on both the skill in which the character is gaining a rank and the class possessed by the character.

Each skill has a skill bonus. The skill bonus is the sum of a number of modifiers. First is the skill category rank bonus, which is derived from the skill rank through the means of a table lookup. To this a pertinent attribute is added (depending on the nature of the skill) along with a class bonus, if any. Finally, any special adjustments are added. All of this work is done during character preparation, so that the overall total is available for immediate use during play.

Conflict System

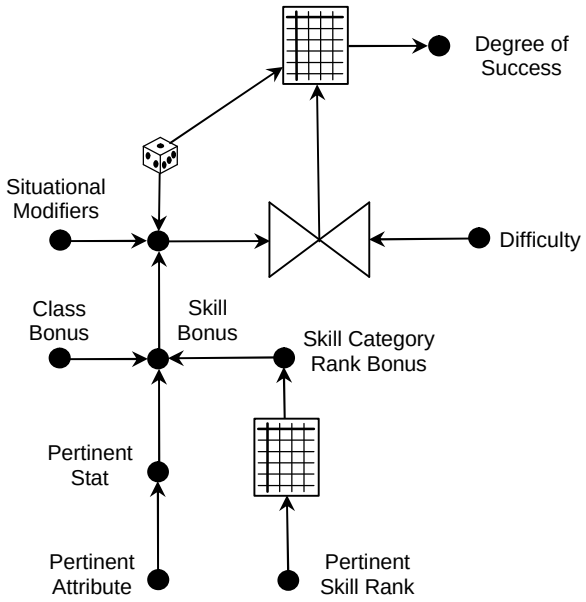


Rolemaster uses classic task resolution. Characters attempt various actions using their skills. All skill rolls use percentile dice (d100) to generate a random number in an "open ended" roll. An open ended roll is performed as follows: Roll the d100 and note the result. If the rolled value fell into the range of 95-100, the dice are rolled again and the results added together. If the second roll again comes up in the range of 95-100, the player keeps rolling and accumulating greater and greater success until a value less than 95 is rolled and the sequence stops. On the other hand, if the initial rolled value fell into the range of 01-05, the dice are rolled again in an open ended fashion, but with the results being subtracted from the character's success instead of added.

To perform contests, the player makes an open-ended roll. To the result is added the character's skill bonus along with any situational modifiers. Finally, to this result is added another modifier based on the difficulty of the task. For easy tasks, this modifier

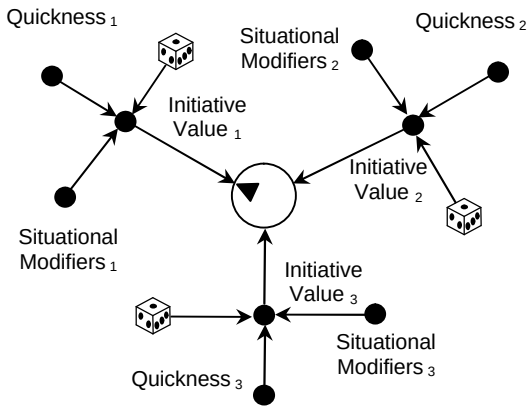
is positive. For difficult tasks, it is negative. Once the overall sum is obtained, the result is determined through a table lookup.

Conflict System (Static Maneuvers)



For combat attacks, the "difficulty modifier" of any attack attempt is the opponent's "Defensive Bonus," or "DB". DB indicates how difficult it is for an opponent to strike a character or monster in combat. It is derived from the Quickness attribute and is modified by the type of armor worn along with any magical adjustments. If a weapon strikes, it delivers damage that lowers the target's hit points and inflicts other effects, such as bleeding and combat penalties of various sorts. The effects of a successful combat attack actually involve two rolls and two table lookups. The first is essentially an attack roll that determines damage and a criticality factor. The second determines any special damage effects based on the criticality factor.

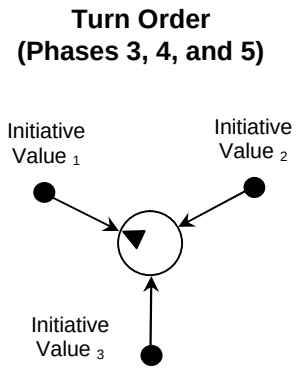
Turn Order (Phase 2)



Turn Order

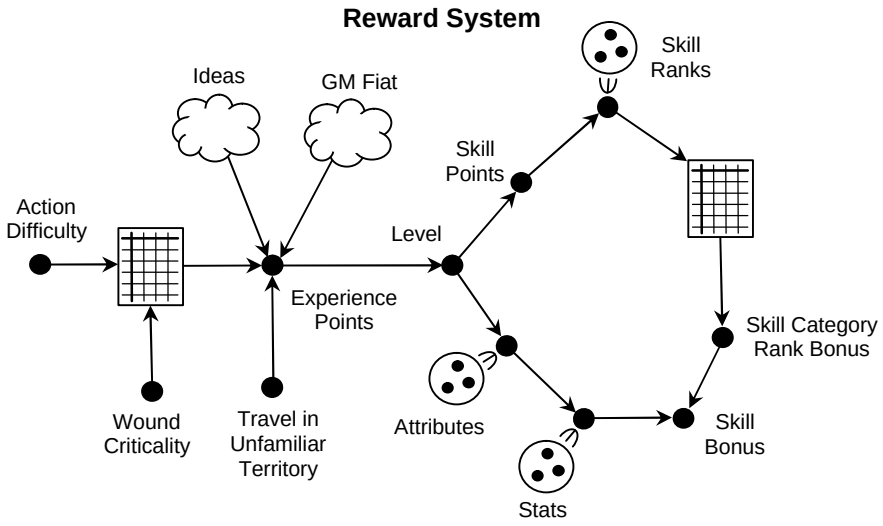
Each combat round is split up into five phases. These phases exist to enable players to try to get their actions in before those of their opponents (with penalties). Conversely, a player may also delay his character's actions in order to let him find the best opportunity to strike. So, players can decide to make "Snap," "Normal," or "Deliberate" actions.

1) Action Declaration Phase - all players declare what actions they are attempting. The declarations are performed in no particular order.
2) Initiative Determination Phase – all players roll 2d10 and add their Quickness bonus. This gives them an Initiative value.
3) Snap Action Phase – resolve all "Snap" actions in the order determined by the Initiative roll. These actions get a penalty.
4) Normal Action Phase – resolve all "Normal" actions in the order determined by the Initiative roll.
5) Deliberate Action Phase – resolve all "Deliberate" actions in the order determined by the Initiative roll. These actions gain a bonus.



Reward System

Characters are awarded experience points for successful actions and for sustaining damage. The amount of experience awarded depends on the character's level and the difficulty of the actions and the criticalities of the wounds sustained and delivered. The values are determined by, you guessed it, table lookups. Experience points are also earned for introducing ideas and concepts that help out on the adventure, for travel in unfamiliar territory, and for whatever else the GM decides to reward. When enough experience points accumulate, the character gains a level.



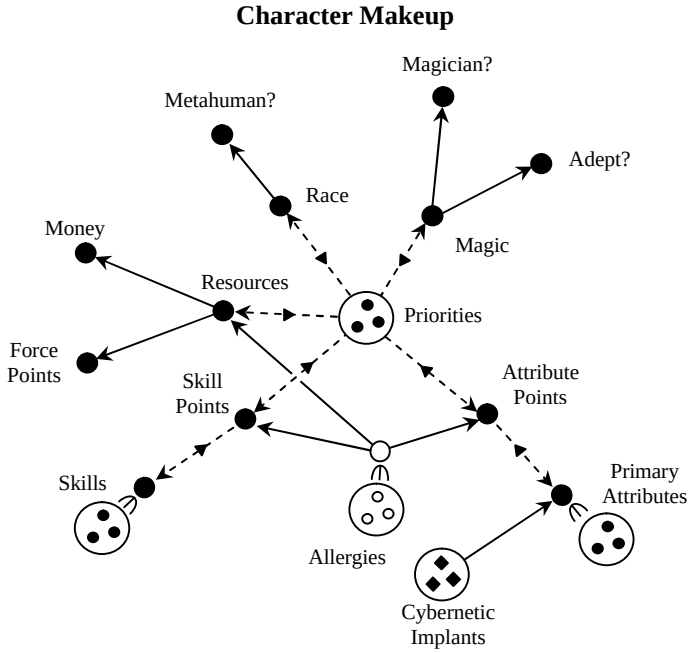
Shadowrun (Second Edition)

Shadowrun is published by FASA Corporation. It is set on Earth in the year 2053, where governments are obsolete and corrupt corporations rule society. Shadowrunners are hired mercenaries enlisted to perform dangerous, illegal, and ethically dubious acts. Such people carry advanced weaponry of various sorts for self defense and enhance their bodies with cybernetic implants. One of the most popular enhancements is the "cyberdeck," which enables a person to mentally hook into the world's computer network, known as the "Matrix." Don't think that *Shadowrun* is a strictly sci-fi game, though. Magic plays as big a role as technology in shaping the world. In the *Shadowrun* future, magic unexpectedly returned, allowing the races of elves, dwarfs, orks, and trolls to arise once again. In addition, shaman, wizards, and other magical practitioners wield strange and potent powers.

RPG Design Patterns Identified

Attribute, Flaws ("Allergies"), Game Master, Generalized Contest, Hit Points ("Condition Monitor"), Last Man Standing, Priority-Grid (character generation), Rank, Resource, Safety Valve (spending Karma points to avoid disaster), Skills, Skill Defaults ("Skill Web"), Success Reward ("Karma"), Template ("Archetypes"), Trauma Gauge

Character Makeup



In creating a *Shadowrun* character, a player must prioritize various character-defining categories from most important to least important. These categories are: "Race," "Magic," "Attributes," "Skills," and "Resources." Based on these priorities, the player is either given options or a quantity of resources from which to draw in each category. For example, any player wanting a non-human character (such as an elf) must specify Race as his highest priority. Players placing more emphasis on attributes and skills gain a larger share of points to use in purchasing ranks in these areas. Players can have their characters start out rich or cybernetically enhanced by placing great importance in Resources.

All characters have the following eight primary attributes: "Body," "Quickness," "Strength," "Intelligence," "Willpower," "Charisma," "Essence," and "Reaction." Magicians get an additional attribute of "Magic." Characters also have an "Initiative"

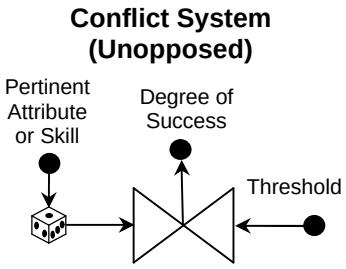
attribute. For most people, this has a fixed value of 1. However, cybernetic implants can raise this value to some degree.

A player can choose to give his character "Allergies" that make him vulnerable to certain substances. Accepting these flaws gives the player more points to spend in attributes, skills, or resources.

If a player is in a hurry, he can base his character on a pre-generated template ("Archetype"). Further character development will then progress at the player's discretion as resources and experience allow.

Conflict System

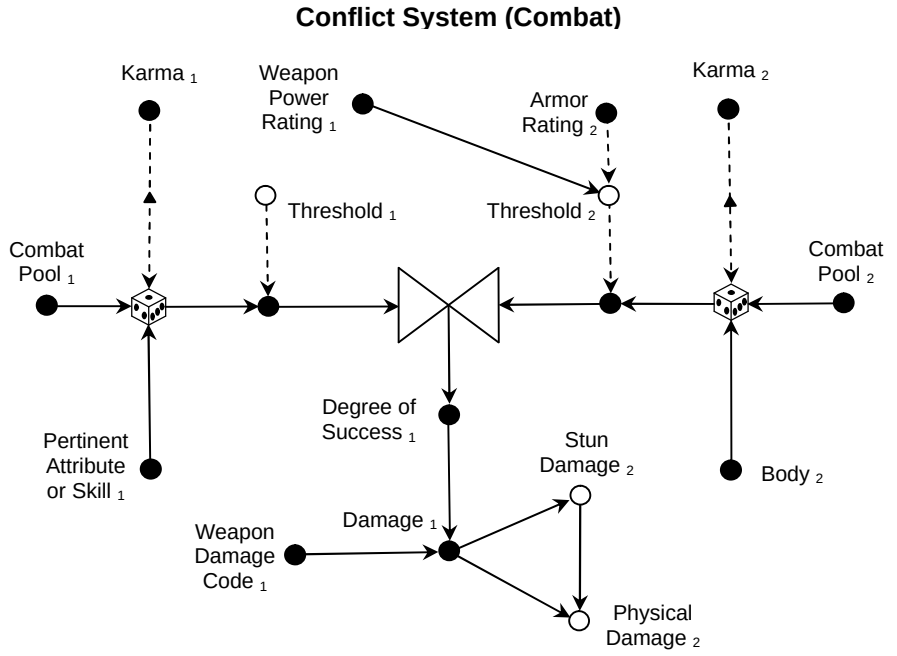
Shadowrun uses d6 dice pools for contests. The player rolls a number of dice equal to his character's pertinent attribute or skill rank. Each die is compared to a threshold, which is given by the game rules for many tasks and is otherwise determined by the game master. Every die resulting in a value greater than or equal to the threshold is counted as a success. Characters need only one success to win. But, the more successes a player rolls, the more advantageous the outcome.



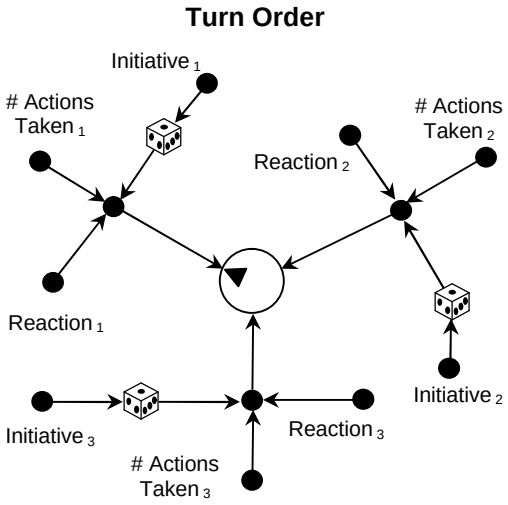
In opposed situations, such as combat, both sides roll and the number of successes on each side are compared. The side with the most successes wins with ties going to the aggressor. The greater the attacker's margin of victory, the greater the effect. For example, the damage level of an inflicted wound increases by 1 for every 2 points of victory over an opponent.

Thresholds generally lie in the range of 1 to 6. But, higher thresholds are possible for exceptionally hard tasks. Beating such difficult obstacles requires an open-ended d6 roll. Players re-roll any d6 resulting in a six and the values are summed. Additional 6s compel further rolls until something other than a 6 is rolled or the threshold has been overcome.

Players can also draw upon their characters' "Karma Pool" to improve their odds in dangerous situations. Players spend Karma Pool points to re-roll dice, avoid disasters, and increase the chances of succeeding in contests. At the end of every encounter, the pool refreshes back to its maximum.



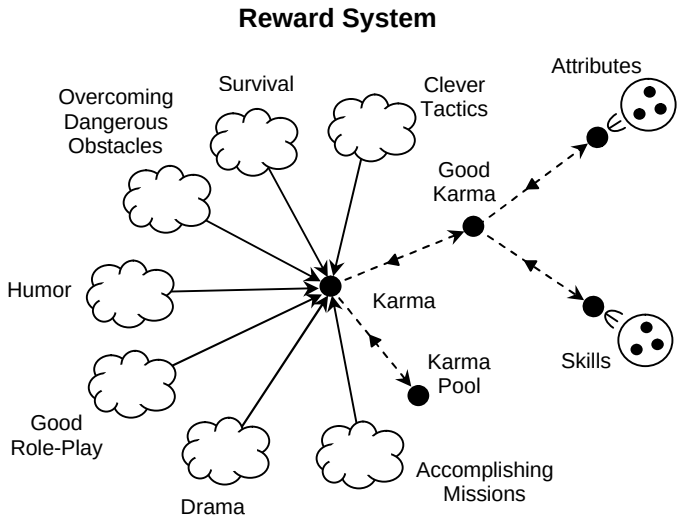
Turn Order



Combat is broken up into discrete rounds, each of which is segmented into an indeterminate number of phases. At the beginning of each round, all players roll a number of d6s equal to their Initiative attribute and sum the results. To this is added the character's Reaction attribute. Each round has a number of phases equal to the highest Initiative total. Play proceeds in order of highest Initiative total to lowest. Every time a character acts, his Initiative total drops by 10. If this leaves the character with a total Initiative greater than zero, he gets another action after any other higher valued Initiative scores are handled. So, if a character obtained an Initiative total of 23, he acts on the 23rd, 13th, and 3rd phase of that combat round. The turn order of characters acting in the same phase is determined by comparing the combatant's Reaction attributes. High score wins.

Reward System

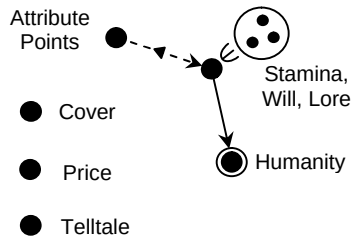
The game master rewards players with experience points ("Karma"). These are given for mere survival, accomplishing missions, overcoming dangerous obstacles, good role-playing, clever tactics, humor, drama, and other things. Ninety percent of Karma goes to "Good Karma" while the remaining 10% goes into the character's Karma Pool. Good Karma points are used to raise attribute and skill ranks.



Sorcerer

Sorcerer is published by Adept Press and was written by Ron Edwards. It is a game that asks the question, "What will you do to get what you want?" In it, players portray sorcerers who have the ability to bind powerful demons to their service. Before play begins, the group must decide what it means to be human. In other words, the gamers must state the aspects of humanity they want to explore in play. Character actions, then, are judged against that standard. Actions that support that vision of decency increase the character's "Humanity" attribute while actions that defy that definition degrade it. Alterations to Humanity are not really rewards or punishments, though: Humanity is conflicted. Sometimes you want a high value and sometimes you want a low value. Low values allow your character to summon demons easily. High values allow your character to banish them. In any case, the Humanity attribute plays a decisive role in the game mechanics, so a character's behavior has a very real impact on his chances for success in various circumstances and, therefore, on the storyline. Needless to say, people whose powers are derived from enslaving demons have ample opportunity to act inhumanely, so the game has a built-in tension between what characters *can* do and what they are *willing* to do. It is perfectly valid to play a Sorcerer that rejects demons and their seductive powers altogether and, instead, play a character with high moral convictions that battles demons and their masters instead. The game mechanics are set up to make both options a viable choice.

Character Makeup

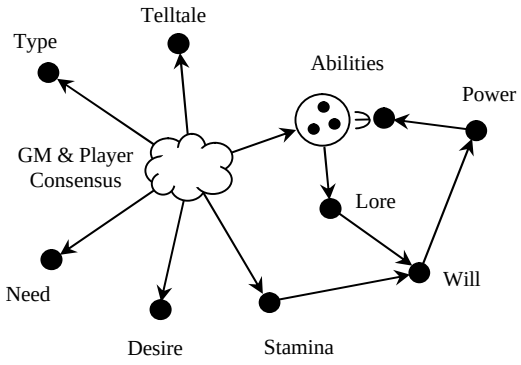


RPG Design Patterns Identified

Attribute, Conflicted Gauge ("Humanity"), Game Master, Idiom ("Humanity"), Narrative Reward, Negotiated Contest, Point Spend Gauge, Resource, Trauma Gauge

Character Makeup

Demon Makeup



Characters have four core attributes: "Stamina," "Will," "Lore" and the all-important Humanity. Stamina deals with physical actions, Will covers mental challenges, and Lore describes the character's knowledge of sorcery. These three are set through the expenditure of a pool of Attribute Points given to players for this purpose. The fourth attribute, Humanity, is set as the maximum of Stamina and Will. As explained above, Humanity provides a gauge of the character's self-image, decency, charity, soul, or whatever aspect of the human state the group wishes to explore. Although Humanity could be classified as following the Idiom pattern, it really is only a pseudo-Idiom. It does translate a character's "morality" into a mechanical effect, but it

doesn't really encourage players to act morally because, at noted above, the attribute is conflicted. So, we cannot consider an increase in a character's Humanity to be a reward (or a punishment, for that matter). In effect, this removes all system-based motives for choosing one way or another, so moral choices become true character-defining decisions.

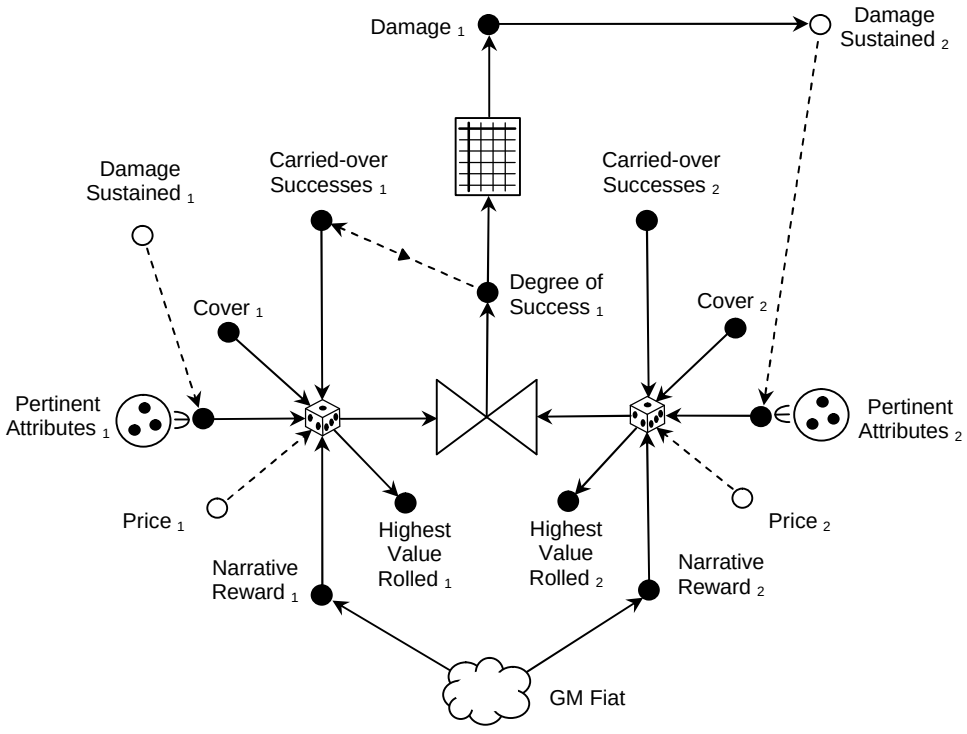
Part of the essential makeup of a *Sorcerer* character is the demon(s) he has bound to his service, since that is a sorcerer's only real source of power. The game is quite flexible concerning the physical characteristics and abilities of demons. If pressed hard enough, a sorcerer's range of powers and potency can rival that of comic book super-heroes.

Conflict System

Sorcerer uses a dice pool system to resolve conflicts. The size of the dice (number of sides) is not important as long as all players use the same sized dice. The number of dice rolled depends on:

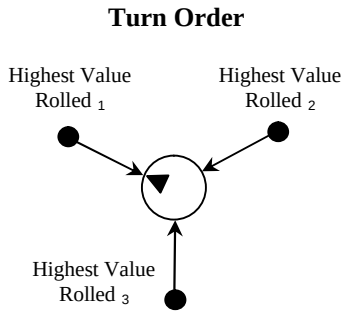
1) The attempted action, which determines what character attributes come into play.
2) Damage sustained, which lowers the size of the dice pool.
3) Temporary modifiers that arise from such things as successes carried over from previous rolls in a sequence of rolls and bonuses for the quality of the player's narration (see the Reward System section below). Interesting and novel descriptions allow players to really heap on the dice, so with entertaining role-play characters can succeed in situations where they would normally fail.

Conflict System (Combat)



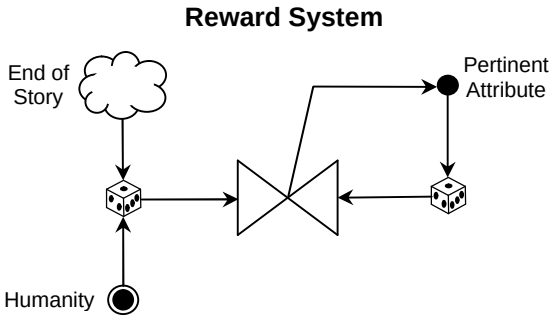
The dice are always rolled in opposition to some other dice pool, either that of another character (player or NPC) or a fixed sized pool depending on the situation. The highest number rolled in either dice pool wins. Note that this is not the *sum* of the dice rolled, but the actual highest number of any single die. If the highest numbers tie, you go to the next highest number to determine the winner. The *degree* of success is the number of dice in the winner's dice pool that have a value greater than the highest number in the loser's dice pool, treating ties as successes. So, if one person rolls a 6, 5, 5, 4, 3, 1 while his opponent roll a 4, 4, 3, 2, 2, the first person wins with three successes. The 6, 5, and 5 are all higher than the highest opponent's die. If one person rolls a 5, 3, 3, 1, 1 and his opponent rolls a 5, 4, 4, 2, the second person wins with two successes. In this case, the 5's tie, so we go to the next highest number. The 4 beats the 3 so the second person wins. The tied fives count toward the second player's successes.

Turn Order



Game flow in a given round starts with players freely discussing what it is that they want their characters to do. Actions can be amended until everyone is satisfied with his declaration. At this point, everyone performing a "proactive" action rolls his dice pool. The order of resolution goes from the highest values rolled to the lowest. So, there is no separate roll to determine action order. If a character has not yet acted, he may choose to abort his pending action in favor of actively defending against another character's action.

Reward System



Bonuses are awarded to individual conflict rolls for colorful descriptions of character actions. These rewards are given in the form of additional dice on the roll, and can accumulate to quite high numbers for narratives of unusual depth and emotion.

At the end of every story, characters have the opportunity to raise one of the Stamina, Will, and Lore attributes. This is done by making a Humanity roll against the score the player wishes to raise. If the roll succeeds, the attribute value increases by one point. If it fails, the player may try again on a different attribute. If all three fail, the character gains nothing.

Torg

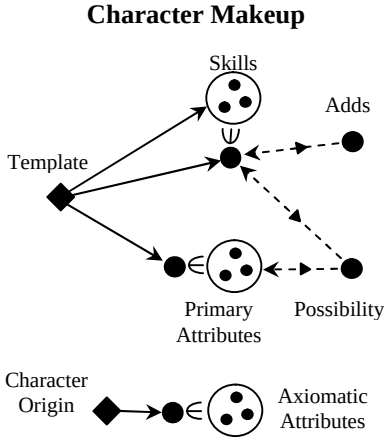
Torg was developed by Greg Gorden and published by West End Games. It is a game set in a multi-dimensional universe in which the various "Realities," or "cosms" (dimensions, universes), are at war. Specifically, modern day Earth has been invaded by no less than seven alternate realities. The various cosms battle over the most precious resource in existence: "Possibility energy." Possibility energy allows the possessor to alter and expand the limits of reality and to shape it to his will. Using various rituals and arcane artifacts, a knowledgeable person can tap into the possibility energy of another world and draw it to himself through an inter-dimensional vortex. The combined energy of a newly discovered, untapped world such as Earth is enough to elevate oneself to the status of a god. Anyone gaining this level of power earns the title of Torg. One being aspiring to do so, known as The Gaunt Man, has discovered Earth's potential and has bent all of his energies on sucking the Possibilities out of Earth to become Torg.

Torg is designed to support cinematic play in realities of various magical, spiritual, social, and technological levels. Earth's reality has been fractured by the intrusion of the invaders' realities. So, some portions of the planet have devolved to pre-historic technology levels, other areas have experienced an infusion of magic that surpasses modern technology in effectiveness, and in yet others technology achieves the level of science fiction.

RPG Design Patterns Identified

Game Master, Generalized Contest, Hit Points, Last Man Standing, Point Spend Attribute, Rank Resource, Safety Valve (spending Possibilities), Template, Trauma Gauge ("Wound Levels")

Character Makeup



Character generation starts with selecting a template from a fairly broad list of character archetypes. These are then customized by the player for his own use by selecting a number of additional skills and spending skill points ("Adds") on them. No more than three points can be spent on any one skill. Each template is required to have one signature skill ("Tag Skill") which must begin with three Adds.

Torg characters have seven primary attributes of "Dexterity," "Strength," "Toughness," "Perception," "Mind," "Charisma," and "Spirit." In order to deal with the extremely wide ranges of capabilities inherent in the different genres the game attempts to handle, the attribute values are gauged on a logarithmic rather than a linear scale. The scale they chose has an attribute's capability increase by a factor of 10 for every 5 point increase. So, a character with a Strength of 11 can lift ten times the weight as a character with a Strength of 6. The normal Earth maximum on their scale for all attributes is a value of 13. Characters also have four "Axiomatic" attributes of Magical, Spiritual, Social, and

Technology. These are fixed to equal the values of the reality from which the character originates.

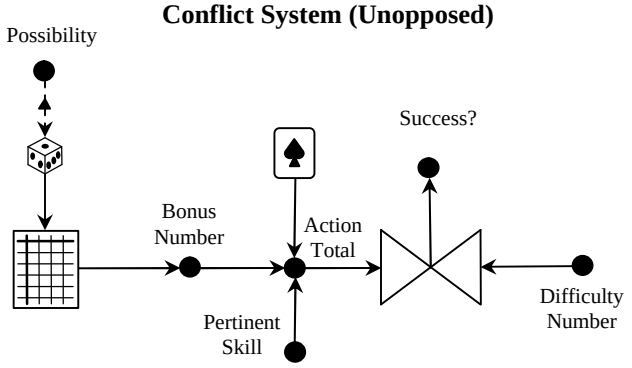
Characters have a Possibility resource. Possibility points are spent on raising skill and attribute values and can be used to alter the outcomes of conflicts.

Finally, characters have a "Shock Damage" resource that acts as a form of hit points and "Wounds" that acts as a trauma gauge.

Conflict System

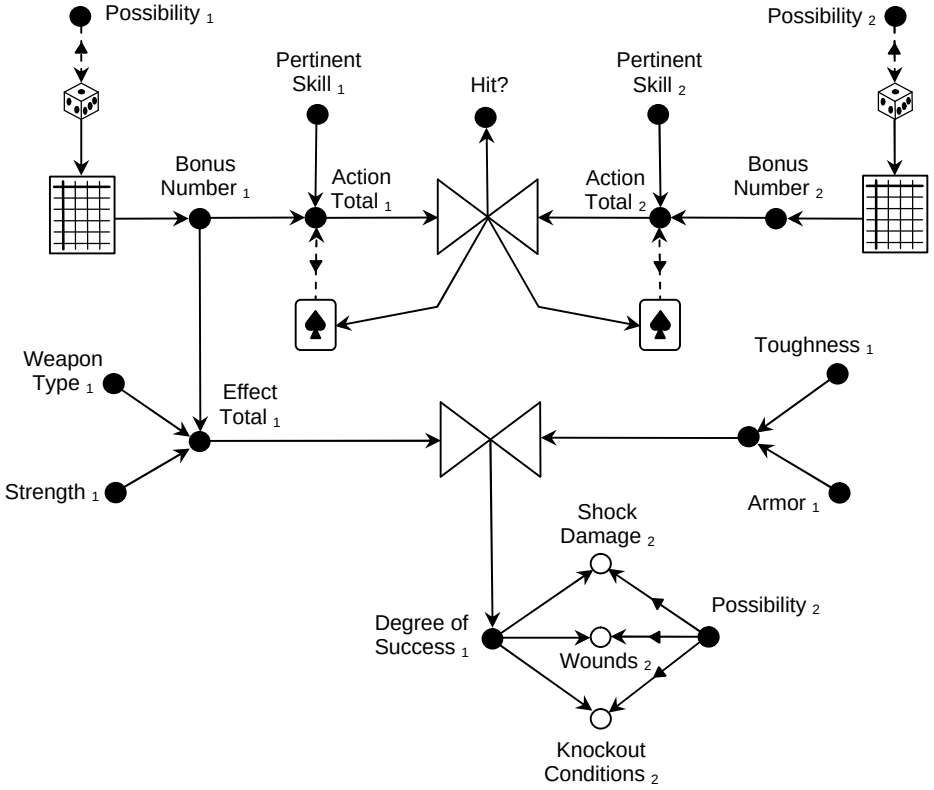
Before contests arise, players are dealt a number of cards from the game's "Drama Deck." Each card lists some special bonus that the player can use in conflicts. "Add +3 to the value of Dexterity, Strength, or Toughness or a related skill" and "May be played as an additional Possibility" are two such examples.

Conflict System (Unopposed)



Whenever a contest arises, the participants state what skills their characters are using in the task. A d20 is then rolled by the aggressor and the result is noted. If a 10 or 20

Conflict System (Combat)



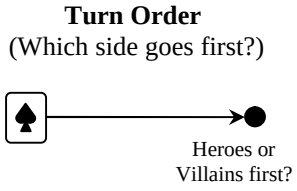
comes up on the die, the player gets to roll again, adding the new result to the previous one. The player continues accumulating more and more success in an open ended fashion as long as 10 or 20 is rolled. The player then has the option of spending a point of Possibility to add another die roll to this total. If he does so, any roll of less than 10 counts as 10 before being added to the running total. This additional roll is also open-ended, so if a 10 or 20 is rolled, the player keeps accumulating more success. (Note: Torg characters are actually segregated into "Possibility-Rated" characters and "Ordinaries". Only Possibility-Rated characters, including all player characters, have Possibility points to spend on re-rolls. Ordinaries do not.)

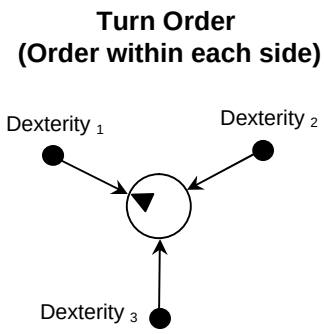
Once this process is complete, the accumulated result is used on a table lookup to determine the characters "bonus number." The character's rank in whatever skill he is using is added to this bonus number to determine his overall "action total." This may be modified by playing any number of pertinent Drama Cards. The action total is then compared to a "difficulty number." If the action total is greater than or equal to the difficulty number, the action succeeds. Some actions, such as combat attacks, require the same roll to be used in determining an "effect total" by adding different numbers (depending on the action) to the bonus number. In this way, a single die roll can be used to determine whether a blow lands and how much damage is delivered.

A character's Shock damage increases every time he is hit, depending on the amount of damage delivered. If the total damage equals or exceeds the character's Toughness, he falls unconscious. Damage is also gauged in the form of "Wound Levels." A Wound is more serious than Shock damage. There are four levels (five if you count "Unwounded"): "Wound," "Heavy Wound," "Mortal Wound," and "Dead." Characters also have "K" and "O" attributes, which are either true or false (circled or uncircled). Some wounds deliver Ks and some deliver Os in addition to Shock Damage and Wound Damage. If a character takes both a K and an O, he is Knocked Out. Given all of this, a blow could result in 2 Wnd K 5, which indicates the character sustains 2 Wounds, a K, and five points of Shock Damage. Once the damaging effects are determined, the target may spend Possibility points to reduce the inflicted damage.

Turn Order

Game flow in Torg is broken down into Acts and Scenes. Scenes are categorized as either Standard or Dramatic. In Standard scenes, action is fast-paced and the Drama Deck cards give an advantage to the players. In Dramatic scenes, action is more drawn out and the villains have the advantage. Initiative for any given Scene is determined by first drawing a card from the Drama Deck. Each card indicates whether the hero or villains go first. All actions of the initiative winners go before those of the losers. The order in which the actions of each side are



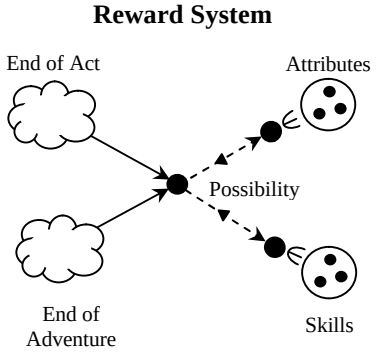


taken are determined by the Dexterity of the various participants, going from highest to lowest.

When the initiative card is drawn from the deck, the card will also specify an "Approved Action." If a player successfully performs an act falling into the categories listed as approved for that round, he gets to draw another card from the Drama Deck to replenish his hand. Otherwise, he doesn't get to draw until the end of the Scene.

Reward System

Torg breaks adventures up into Acts. At the end of each act, the Game Master awards each player zero to three Possibilities to add to his character's pool. At the end of an adventure, the Game Master awards another 6 to 12 Possibilities. These can be saved for later use on improving success in conflicts or can be spent immediately to raise attribute values and skill ranks.



Universalis

Universalis (pronounced “universe-alice”) was written by Ralph Mazza and Mike Holmes and is published by Ramshead Publishing. It is a storytelling game that so defies common role-playing game design conventions that some people may balk at calling it a role-playing game at all. *Universalis* has no Game Master. Or, from another perspective, in *Universalis* all players are the Game Master. Players can create characters at will, introduce new plot elements, and even seize control of other players’ characters. Nevertheless, it does incorporate what the author considers the essential element of role-playing: players portray the roles of characters. If you’re playing a role, then you’re role-playing. That concept seems pretty straightforward. And yet, *Universalis* breaks all the rules. At the same time, it adopts some easily recognizable design patterns of more traditional role-playing games.

Universalis provides no setting and is not geared toward any particular story genre. Everything is left up to the players to create, with some rules governing how and when story elements are introduced. There are even rules that govern how you can change the rules if the players feel the urge.

The game sets up a simple economy through the distribution and use of “coins.” Facts are introduced into the world by spending these coins. The more coins a player has, the more potential he has in introducing new facts into the story. All players start with an equal number of coins. More are distributed at the end of every scene. Coins are also earned by introducing conflicts into the plot. This basic economy keeps any one player from dominating the story to the exclusion of others. It ensures that everyone gets a voice. It also encourages players to introduce facts that are harmful to their characters (and others) in order to earn more coins. This design element is significant, because all interesting stories involve some form of contention or dispute. Any storytelling game lacking a Game Master needs some well crafted way to ensure conflicts arise.

The game starts with a clean slate. There is no prep-work to playing *Universalis*. Play begins with players spending coins to set up the basic “Tenets” of the game. For example, one group playing the game set up the following Tenets (each costing one coin each): “The story is a mystery,” “It is set in cave-man times,” “No dinosaurs,” “The mystery involves why no buffalo have arrived this year,” “Our meat stores are low,” “The story is set in the prairie,” “We are tired of eating prairie dog meat,” “Our tribal totem is the buffalo,” “Our enemies are the Crow totem people.” In setting up these tenets, only the “No dinosaurs” tenet was disputed between players. One player wanted dinosaurs while another did not. Most players did not care. The conflict resolution rules handled the dispute smoothly with the “No dinosaurs” side winning the contest.

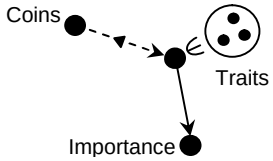
RPG Design Patterns Identified

Attribute (“Importance”), Currency, Dice Pools, Negotiated Contest, Rank, Resource (“Coins”), Trait

Character Makeup

All characters have the single attribute of "Importance." Importance gauges how important the character is to the story. It has a numeric value equal to the number of coins spent on developing and equipping the character. To permanently remove a character from play, a number of coins must be spent equal to its Importance rating. Note that this does not mean the character is killed. A character can cease to appear in scenes without requiring his death. Conversely, a dead character can re-appear in later scenes as a ghost or in flashbacks.

Character Makeup



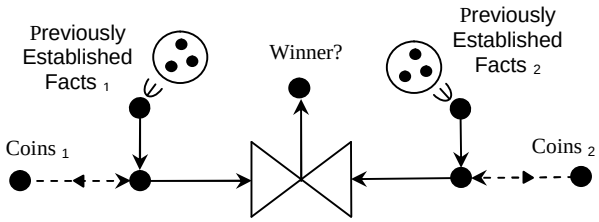
Characters also have traits. Traits are abilities and other characteristics established as facts through the expenditure of coins by the players. Even a character's name is a trait that must be purchased. So are physical wounds.

Traits are ranked. Every coin spent on a trait increases its rank by 1. This, in turn, increases the character's Importance rating. So, if a character sustains a rank 2 wound to his thigh, his Importance rises by 2 which actually makes it harder to eliminate the character from the story.

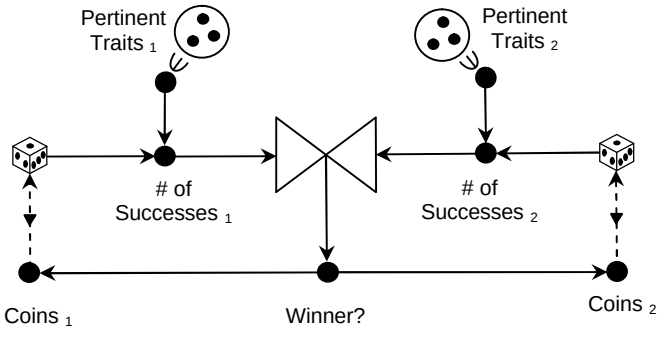
Conflict System

Disputes between players are resolved in two phases: "Negotiation" and "Bidding." During Negotiation, the players involved in the challenge try to reach some mutually agreeable solution. If this can be accomplished, play then proceeds without further interruption. However, if an agreement cannot be reached, the conflict proceeds to the Bidding process. The challenger must bid at least one coin. If he does not, he automatically loses the dispute. Assuming the challenger bids at least one coin, the bidding process opens up for all players to contribute to either side. Any previously established facts that are contradicted by either side can be used as additional leverage in the bidding process. The side with the highest final bid wins.

Player Conflict System



Character Conflict & Reward System

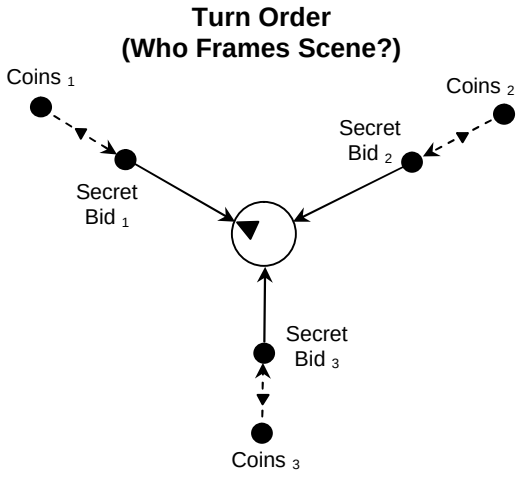


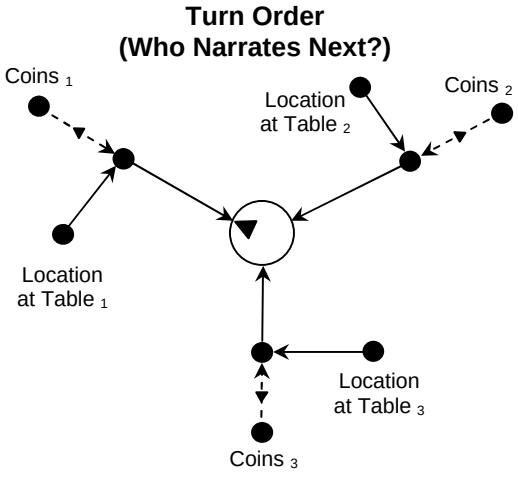
If a player introduces a dispute between a component he controls and a component under the control of someone else, a conflict arises. Each side in the conflict builds a dice pool of d10s. All players can contribute to the dice pools. Dice are added to pools by either drawing on a character's traits (trait rank = # of dice) and/or buying dice by spending coins (coins spent = # of dice). Traits can only be applied if they are applicable to the confrontation. Once the dice pools are established, both sides roll. All dice rolling 1 to 5 are considered successes, the rest are discarded. The side rolling the greatest number of successes wins. Both sides contribute to the description of the outcome. Naturally, the winner narrates first and the loser must accede to the winner's previously established facts.

Turn Order

Before each scene, each player makes a secret bid to frame the next scene. The highest bidder gets to describe where the next scene takes place and what characters are initially involved. The coins bid by the winning bidder are used by him to purchase facts in framing the scene. The framer can describe the scene without interruption until the scene's location and time are determined and at least one component (character) appears.

Turn order normally happens in a clockwise fashion from player to player. A player's turn ends when he either wants it to or he can no longer afford to introduce new story elements due to a shortage of coins. However, any other player can "Interrupt" another player and seize control of the storyline by spending one coin. Play then proceeds in a clockwise fashion from the interrupting person after he is done.





Reward System

In any conflict involving opposed dice pools, the contest actually ends up generating coins for both sides of the conflict. So, while characters can lose out in disputes,

players only benefit from them. The winner of the conflict adds up the results of all dice that were successes and gains a number of coins equal to that value. The loser gains a number of coins equal to the total number of dice he rolled in the conflict, including both successes and failures. Both the winner and loser buy facts detailing the outcome of the struggle equal to the number of coins generated by the dice. However, both sides then get to keep their newly generated coins to replenish their pools. So, the game essentially rewards conflict.

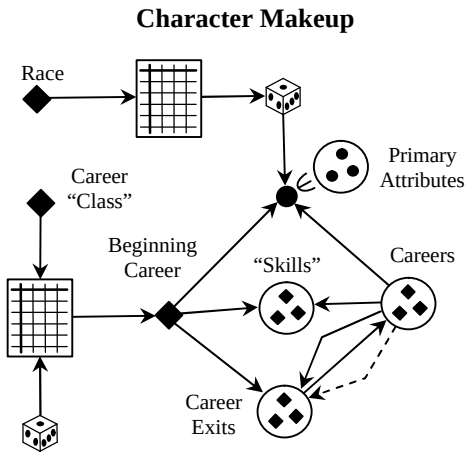
Warhammer Fantasy Role Play

Warhammer Fantasy Role Play is published by Games Workshop Ltd. It is a traditional fantasy game that pits the forces of Law against those of Chaos with the heroes getting stuck somewhere in the middle. The main rulebook includes a broad overview of the "The Known World," whose map is obviously a distortion of Earth's map. It contains a number of lands, including "Albion" (England), "The Old World" (Europe), and "The New World" (North America). Player characters generally come from somewhere in "The Old World."

RPG Design Patterns Identified

Alignment, Attribute, Class Tree, Experience Points, Faction (Law vs. Chaos), Game Master, Generalized Contest, Gift, Hit Points, Random Attribute, Safety Valve, Skill

Character Makeup



Warhammer has a Tolkienesque setting allowing players to choose from the standard fantasy races of "Man," "Wood Elf," "Dwarf," and "Halfling." Once this is done, the player selects an alignment for his character from the choices of "Chaotic," "Evil," "Neutral", "Good", and "Lawful". In some places, the game text states that the alignment associated with a race must be taken by the player for his character. At other points, though, the text states that alignment is a choice. Due to these contradictory statements, there is little doubt that the majority of players end up choosing the alignment they want to play.

Once the race is chosen, a character's 13 primary attribute values can be determined. The various attributes are: "Movement" (M), "Weapon Skill" (WS), "Ballistic Skill" (BS), "Strength" (S), "Toughness" (T), "Initiative" (I), "Attacks" (A), "Dexterity" (Dex), "Leadership" (Ld), "Intelligence" (Int), "Cool" (Cl), "Willpower" (WP), and "Fellowship" (Fel). Characters also have a "Wounds" (W) resource that acts as a form of hit points. The rules contain a table indicating what dice and formulas should be used for calculating the attributes. So, the Ballistic Skill of an Elf is "2d10+20" while the Strength of a Man is "d3+1". Some attributes end up having values lying in the range of 2 to 50 while others range from 1 to 4. In fact, "Attacks," an attribute indicating how many attacks a character gets in a round of combat, always starts at 1. There is no absolute uniformity of number range from one attribute to the next. However, many of them are used in contests by comparing the number to a roll of d100. Those that are used in this fashion fall into the 2 to 50 range initially.

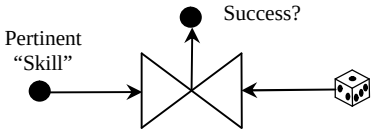
Characters also have a resource knows as "Fate Points." Fate Points follows the Safety Valve Design Pattern in preventing premature character death.

Once the attributes and Fate Points are determined, the player selects one of four "Career Classes" for his character from the following list: "Warrior," "Ranger," "Rogue," and "Academics." The "Career Class" is actually a class category, because each category is associated with a table that is used by the player to randomly determine his character's "Career." It is the randomly selected "Career" that actually follows the Class Design Pattern. The number of class possibilities is impressive. It includes entries as wide ranging as "Alchemist's Apprentice," "Beggar," "Initiate," "Mercenary," "Rat Catcher," and "Squire." Each class gives a list of gifts ("Skills") along with chances to obtain them, equipment, attribute adjustments that characters can earn as they progress in the career, and "Career Exits." The career exits enable a character with a given class to advance to some other class. When he does, his loses his old career exits and gains a new set of career exits.

Warhammer "skills" actually fit this book's definition of "gifts," since no ranks are ever gained in them and they do not improve as play progresses. (The attributes on which they are based do improve, however.) Their list of gifts is quite lengthy. It includes options such as "Boat Building," "Gamble," "Mining," "Night Vision," and "Strike Mighty Blow."

Conflict System

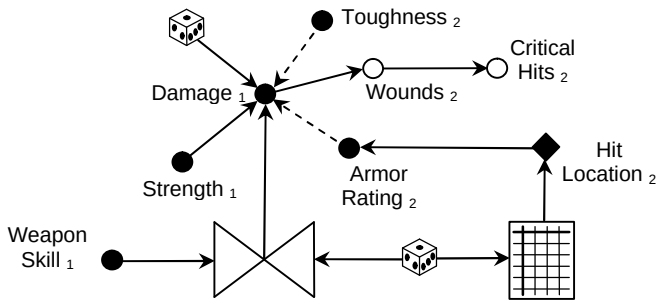
Conflict System ("Skill" Use)



Contests are performed by first determining whether a character has the gifts necessary to attempt an action. If so, the player rolls d100 and compares the result to his character's pertinent attribute. If the result is less than the attribute value, the task succeeds.

In combat, the pertinent attribute to use in attacking an opponent is "Weapon Skill." If the hit succeeds, damage is determined by rolling a d6 and adding the aggressor's Strength. The target's Toughness attribute is then deducted from the result to give a damage amount. The hit location is then determined by reversing the numbers rolled on the d100 and performing a table lookup (i.e., an attack roll of 83 becomes a hit location of 38 on the table). The armor rating of the struck body part is subtracted from the damage to give a final damage total. This damage value is subtracted from the character's hit points ("Wounds").

Conflict System (Combat)



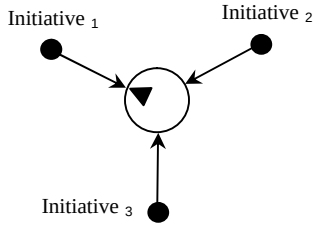
If the d6 used in the damage roll results in a 6, the player has a chance for additional damage. To do so, he must roll another d100 and compare it to his "Weapon Skill"

attribute. If the roll is once again less than this value, the player adds another d6 to his damage. If a 6 is rolled again on the d6, additional d6s are accumulated as long as 6s are rolled in an open-ended fashion.

Warhammer hit points work in an interesting way. When a character's "Wounds" resource drops to zero, the character does not fall. Rather, he finally starts taking real wounds. That is, he starts accumulating "Critical Hits" that have serious consequences to the character's health such as incapacitation or death. These are determined by cross indexing the inflicted damage with another percentile roll. This, along with the hit location, determines the actual wound delivered. Severed limbs, broken bones, and death are common results.

Turn Order

Turn Order

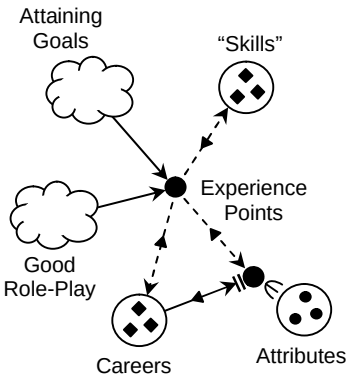


When opposing forces first meet, anyone unaware of the opposition is automatically surprised and cannot attack on their first round of combat. Otherwise, action order goes from the character with the highest "Initiative" attribute and proceeds sequentially on down to the character with the lowest Initiative value. So, initiative is purely Karma-based.

Reward System

Warhammer rewards players by giving their characters experience points. Experience points are awarded for attaining an adventure's goals and "good role-playing." These points can be spent to buy new gifts and to raise attribute values. These are limited, however, to the "advance scheme" of the current class. Once the maximum advancement for an attribute has been attained in a class, the player cannot raise that attribute further until he progresses to the next class, which must be one of the available "career exits" listed in the class description. Any player wishing to have his character advance to another class must expend experience points to do so.

Reward System



The World of Darkness

The World of Darkness is published by White Wolf Game Studio. The game consists of a core rulebook and an array of supplemental materials, including *Vampire: The Requiem*, *Werewolf: The Forsaken*, and *Mage: The Awakening*. In White Wolf's previous releases, *The World of Darkness* was really a common setting shared by similar but independent game systems. The latest release unifies all of the old disparate rules under one umbrella. The setting is modern Earth with a heavy gothic horror theme throughout. Players generally portray either mages or supernatural predators, such as vampires and werewolves, whose quarry includes the unsuspecting public. Although players can take ordinary mortals as characters, the game emphasizes the fantastic. At the time of this writing, *Vampire: The Requiem* is the only major supplement available for the latest release. Much of the mood and theme of the planned supplements can be surmised from the game's previous incarnations, but please understand why this description focuses primarily on vampires.

RPG Design Patterns Identified

Attribute, Conflicted Gauge ("Blood Potency"), Faction (Clans and Covenants), Flaws ("Derangements"), Game Master, Generalized Contest, Gifts, Hit Points ("Health"), Idiom ("Morality"), Last Man Standing, Point-Spend Gauge, Priority-Grid (character generation), Skills, Template, Trauma Gauge

Character Makeup

The World of Darkness characters have nine primary attributes: "Intelligence," "Wits," "Resolve," "Strength," "Dexterity," "Stamina," "Presence," "Manipulation," and "Composure." These attributes are related to one another as shown in the following table:

	Mental	**Physical**	**Social**
Power	Intelligence	Strength	Presence
Finesse	Wits	Dexterity	Manipulation
Resistance	Resolve	Stamina	Composure

So, mental actions involve Intelligence, Wits, and/or Resolve; Physical actions use Strength, Dexterity, and/or Stamina; Social actions utilize Presence, Manipulation, and/or Composure. Attributes listed on the "Power" row represent a level of raw force on which the character can draw in each of the three areas. Finesse attributes represent the character's flexibility and quickness in a given area. Resistance conveys the character's mental, physical, or social toughness.

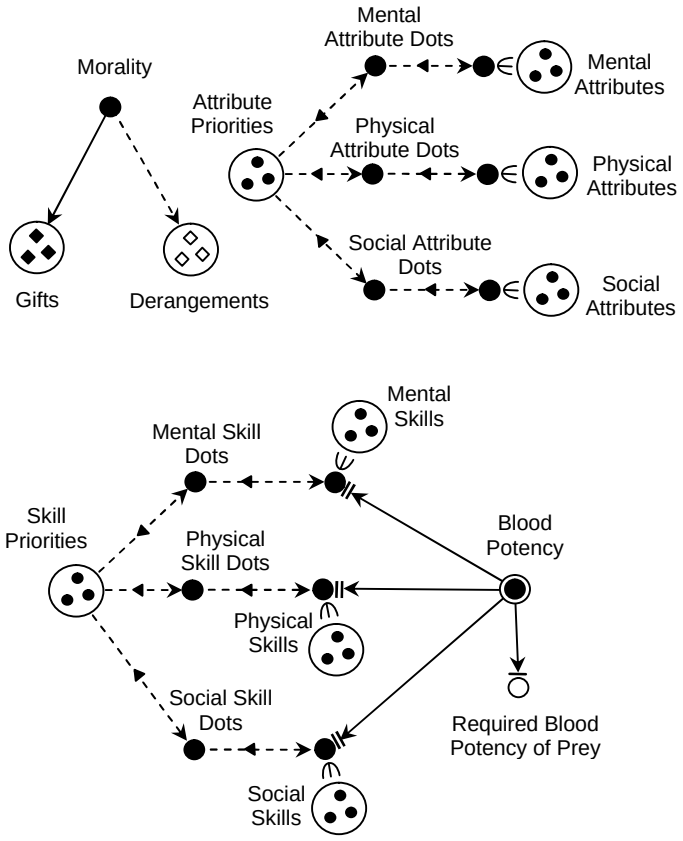
Attributes are generally rated between 1 and 5 "dots," although supernatural characters can have substantially higher values. These ratings are initially purchased by spending three pools of dots, one for each of the Mental, Physical, and Social categories. The player prioritizes these categories: the highest priority category gets a large quantity of "dots" from which to draw while the lowest priority gets a small amount.

Characters have gifts and skills that are similarly broken up into the categories of "Mental," "Physical," and "Social." Purchasing skill ranks and gifts is analogous to buying attribute values in that players prioritize the three ability categories. The more a player emphasizes a category, the greater the number of "dots" he gets to spend in that category. Like attribute values, skill ranks range in value between 1 and 5.

Each Vampire character is a member of one of five "Clans" and one of six "Covenants". The various Clan choices are "Daeva", "Gangrel", "Mekhet", "Nosferatu", and "Ventrue". The Covenant choices are "Carthian Movement", "Circle of the Crone", "Invictus", "Lancea Sanctum", "Ordo Dracul", and "Unaligned". Ostensibly, a character's Clan is determined by the bloodline of the vampire that spawned him. A Covenant is essentially a political movement or organization. In design pattern terms, a character's Covenant is a Faction and his Clan is both a Faction and a Template (since it determines a character's starting skills).

Vampire characters also have an attribute known as "Blood Potency." The character rating in this attribute limits the ranks he can attain in various vampire-related skills ("Disciplines"). However, a vampire's Blood Potency value also determines the minimum Blood Potency rating of his prey. Highly potent vampires demand highly potent sustenance. Characters with a Blood Potency of 1 can feed on the blood of animals. A higher Blood Potency forces a vampire to prey on humans or possibly even other vampires. So, there exist valid reasons for characters to want both high and low Blood Potency ratings at different times. Therefore, Blood Potency is strongly conflicted (and, subsequently, quite interesting from a design standpoint).

Character Makeup

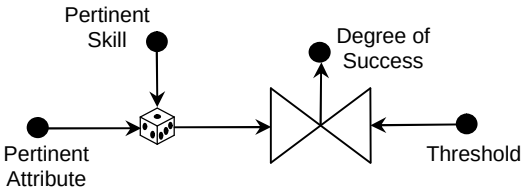


Finally, characters have a "Morality" attribute ("Humanity" in *Vampire: the Requiem*) that follows the Idiom pattern. The attribute gauges the character's soul, his sense of "right" and "wrong," by the acts he performs. Selfish or evil acts tend to degrade Morality while charitable and trustworthy acts raise it. (In *Vampire: the Requiem*, charitable acts only allow the player to spend experience points to raise Humanity if he chooses. There are no "free" Humanity gains once you become a vampire.) When a character's Morality drops, he may experience a "Derangement," such as "Depression," "Paranoia," or "Hysteria." Even more severe consequences are possible as well. In *Vampire: the Requiem*, if a vampire's Morality drops to zero, he becomes an uncontrollable raving bloodthirsty monster that is no longer playable as a character. High Humanity provides benefits as well. A vampire in touch with his human side can stay awake after sunrise longer and mingle with mortals more easily.

Conflict System

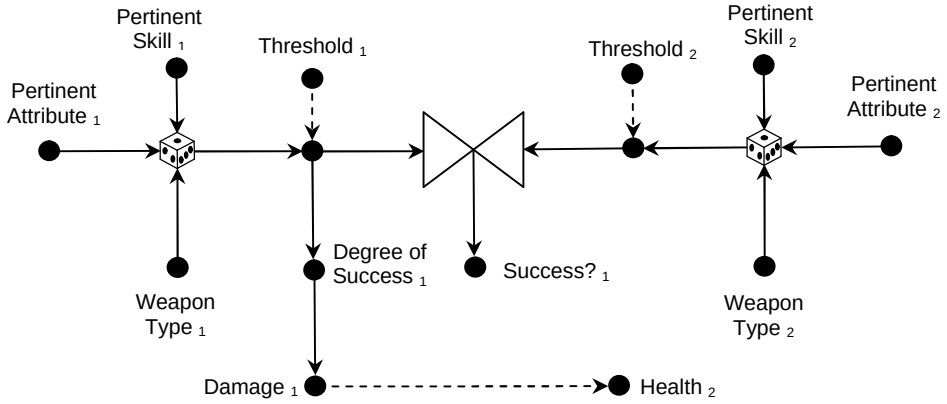
Contests in *The World of Darkness* are performed using d10 dice pools. The number of dice rolled is usually the sum of an attribute value and a skill rank or, in cases where skills are not pertinent, the sum of two attributes. This number can be modified by the game master depending on circumstance. When rolled, any dice coming up with a value of 8 or more count as successes. Any die rolling a 10 is rolled again in an open-ended fashion to potentially generate even more successes. In general, only one success is needed to accomplish at a given task. But, multiple successes indicate that the character triumphs in a spectacular way. The more successes generated by a roll, the greater the character's mastery of his endeavors.

Conflict System (Unopposed)



If a character's action is directly opposed by the actions of another character, both sides roll as described above. The character obtaining the greater number of successes wins. The magnitude of his victory is not determined by the difference between the two rolls, however. All of the victor's successes are counted in determining the conflict's outcome. The loser's successes are discounted.

Conflict System (Combat)

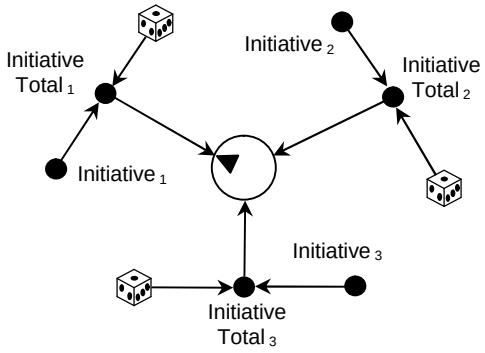


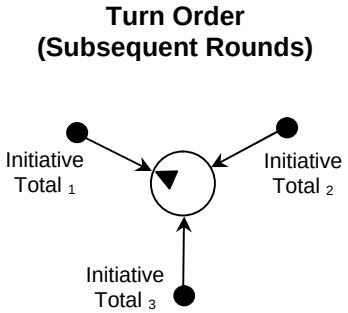
Depending on type, weapons give bonuses to the wielder's dice pools in melee. The successes of attack rolls directly translate into damage on a point-per-point basis. Damage counts against the target's Health attribute, which follows both the Hit Points and Trauma Gauge patterns. Damage is divided into three types: Bashing, Lethal, and Aggravated. Bashing and Lethal are ordinary forms of damage that can be healed with relative ease through supernatural means. Aggravated damage cannot be healed so easily and is inflicted in different ways on different characters. For examples, silver weapons deliver aggravated damage to werewolves. Vampires are similarly vulnerable to fire.

The game master can call for an extended roll in situations where he feels a task will take a while to accomplish. In such cases, he comes up with a total number of successes he believes the task requires and allows the player to accumulate successes through a number of rolls. He will also generally state how much game time each roll represents and may decide that the character has a limited amount of time in which to attain victory. If the player obtains the requisite number of successes within the specified number of rolls, the action succeeds. Otherwise, it fails, but can sometimes be re-attempted at a later date with penalties.

Turn Order

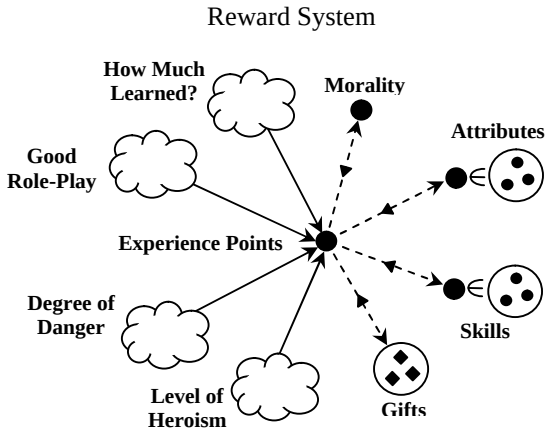
At the beginning of a scene involving conflict, players roll a d10 and add their characters' "Initiative" attributes to determine the action sequence. Players take turns in order of highest initiative total to lowest, at which time the sequence repeats (without a re-roll) until the conflict ends.





Reward System

The World of Darkness rewards players with experience points. The amount is generally between one and five points per



session. The size of the award is determined by the game master based on the quality of role-play, the amount he believes a character would have learned from an adventure, the degree of danger involved, and the level of heroic performance. One point is awarded for just showing up to the game. Experience points can be spent on obtaining new gifts, gaining ranks in skills and attributes, and raising "Morality."

Appendix A: Design Pattern Ideas

This appendix is really just a scratch-pad containing some concepts noted during the game study that could potentially be written up as Design Patterns in the future. They are listed here for one or more of the following reasons:

1) It is an interesting tidbit of game design, but only a single reference to the concept has been found and we are looking for another instance before we can call it a pattern.
2) I haven't decided that the pattern is worthy of a write-up on its own, and am hoping to discern some more general pattern that incorporates the concept in the future.
3) I plan to do a write-up, but just haven't gotten around to doing so yet (possibly because I haven't figured out anything intelligent to say about it).
4) I've already done a write-up on it, but forgot to remove it from this list.

Assist: Multiple characters help out on a roll – InSpectres' Teamwork – HeroQuest "Augmenting." Most games have this Pattern.

Balancing Loop: A Balancing Loop exists in a contest if some gauge that feeds into the contest is altered in such a way that the gauge either gains a Failure Reward or a Success Punishment. If the gauge is conflicted, this simply means that the gauge is altered in such a way that future contests are more likely to turn out differently (i.e., more likely to fail if the contest succeeded or more likely to succeed if the contest failed). Capes rewards "Story Tokens" to the losers of conflicts. These are used in future contests to introduce new characters and to buy extra actions during conflicts, increasing their chances of winning those contests. Dogs in the Vineyard inflicts "fallout" when a character

Cascading Gauge: A gauge value has a maximum. If more is added to it, the gauge "overflows" into another gauge. – Fudge and Shadowrun Wound Levels (the "more detailed method" in section 4.57) – HeroQuest "Masteries."

Cause-First Abilities: Ability mechanics are custom designed to generate a desired output based on what the author feels is "realistic" given the game's premise - D&D, RIFTS, Warhammer. That is, a standard cause/effect relationship is established, where a given cause produces a given effect. Cause-First Abilities are in contrast to Effect-First Abilities.

Cause-First Contests: The end results of contests are determined purely by what the game designer considers to be "realistic" based on the contests' various contributing factors without regard to the implications of those effects – D&D, RIFTS, The World of Darkness.

Confessional Mechanic: InSpectres (Also, check out *Shadows in the Fog*).

Conflicted Resource: can spend up in one fashion and spend down in another.

Damage Resistance: aka Absorption, Toughness – GURPS, Hero System 5th Edition, TORG "Toughness," Werewolf "soaking" damage - as opposed to damage ablation, per say Hero System? Or deflection per D&D (or both of these per GURPS)? Using "Damage" in the name makes it too specific to Hit Points, I think. In any case, all of these examples can probably be adequately covered in my write-

up of the Gauge Design Pattern merely as ways in which one gauge can be used to support or detract from another. So, this pattern will probably be pruned.

Death Spiral: a contest failure result that has the effect of making it more likely that future contests will fail – Shadowrun Wound levels.

Degree of Success Contest: a contest where a graduated scale of success or failure is generated rather than a simple win/lose result – HARP, HeroQuest, InSpectres, Marvel Super Heroes, Paranoia xp, The Riddle of Steel, Rolemaster, Shadowrun, Sorcerer, TORG, The World of Darkness.

Diceless: No fortune-based contests appear anywhere in game -- Amber, Nobilis, Puppetland.

Dice Pools: Sorcerer, The Riddle of Steel, Universalis, The World of Darkness, Shadowrun.

Drama-Based Contest: Nobilis on initiative, Puppetland on both Initiative and Conflict Resolution.

Effect-First Abilities: The in-game effects of abilities are designed to fit the mechanical output of the contest resolution rules – Hero System 5th Edition powers, Universalis, Dogs in the Vineyard – any trait-based game. Effect-First Abilities first establish the desired outcomes, or effects, of an action without regard to their cause. Once that is determined, an appropriate cause is made up to rationally explain how that outcome was actually achieved.

Effect-First Contests: The end-results of contests are narrated to fit the mechanical output of contests. In other words, situation is modified to fit the results of a contest – My Life with Master, Dogs in the Vineyard, InSpectres (?), Ars Magica magic.

Fortune-Based Contest: Sorcerer conflict resolution, My Life with Master conflict resolution, D&D task resolution, Rolemaster task resolution.

Fortune in the Middle: Sorcerer, My Life with Master.

Fortune at the End: D&D, RIFTS.

Flaws: aka Faults – Fudge, The Riddle of Steel, Nobilis - Restrictions

Free and Clear Initiative: everyone states their actions and can modify them until everyone is satisfied with their declarations, then initiative is rolled (is this just "Drama-Based"?) - Sorcerer

Gambled Resource: Donjon – Wealth, Sorcerer - Humanity (?), HeroQuest – Advantage Points.

GM-less: Universalis, Capes, Discussed as an alternate, or "advanced", method of playing Amber.

Hacking: Code of Unaris.

Hot Potato Initiative: hand off story telling control to another player based on specific criteria – used in ThemeChaser on the Forge.

Karma-Based Contest: Code of Unaris conflict resolution, Nobilis conflict resolution, Universalis conflicts between players is a bidding process, Call of Cthulhu. initiative uses DEX from highest to lowest & roll d100 in case of tie

Logarithmic Scale: Fudge, TORG, Hero System, HeroQuest?.

Margin: The Riddle of Steel, Sorcerer, Shadowrun.

Margin Rollover: Sorcerer, Werewolf: The Apocalypse on Damage for Firearms (pg. 227).

Megalith: Game designed with a core rulebook to be endlessly expanded with supplements. D&D (d20), Rolemaster, The World of Darkness, TORG.

Monolith: Game designed as single rulebook. Sorcerer, Donjon, My Life with Master.

Narrative Control Contest: a contest where players vie for the right to narrate the outcome of a conflict - Capes, Code of Unaris, Donjon, InSpectres, Universalis.

Never Ending Story: Any game that does not come to a definite ending point.

Nonlinearity: The game abruptly changes in some fashion – My Life with Master has formulas that trigger specific events; The Pool has players gamble dice from their pool so a character can instantly go from being powerful to being powerless.

"One Shot" Game: Great Ork Gods.

Open-Ended Roll: Rolemaster, TORG, The Riddle of Steel, Warhammer – Damage.

Opposed Rolls: The Riddle of Steel – contested rolls, My Life with Master

Purchased Events: Universalis.

Race: aka Species. Similar to Class and Template patterns: D&D, HeroQuest, Warhammer, Rolemaster (isn't this really just a class?).

Recycled Fortune: Use a dice roll in one fashion by looking at it one way, then use it for another purpose by looking at it in another – Elfs uses the same roll of a dice pool to determine Initiative and Conflict Resolution; Sorcerer the same roll of a dice pool to determine Initiative and offensive side of Conflict Resolution; TORG uses d20 roll for both success and effect by adding different values. Warhammer – Attack Roll and Hit Location.

Refresh: A vital game resource is "re-fueled" periodically to ensure smooth game flow.

Reinforcing Loop: A Reinforcing Loop exists in a contest if some gauge that feeds into the contest either gains a Success Reward or a Failure Punishment. If the gauge is conflicted, this simply means that the gauge is altered in such a way that future contests are more likely to turn out with a similar result (i.e. more likely to succeed if the contest succeeded or more likely to fail if the contest failed).

Relationship Map: Sorcerer

Resource Refreshing: A resource is periodically "refreshed" by giving it more resources to allow the game to flow. It acts as a sort of "fuel." HP in HeroQuest.

Rolled Initiative: Donjon, D&D, Rolemaster, Shadowrun.

Randomized Resource: Rolemaster – resource for setting attributes, Great Ork Gods – resource for setting attributes, starting Gold Pieces in D&D.

Round Robin Initiative: D&D, HeroQuest.

Shared Gauge: a gauge whose value is shared by all characters – My Life with Master: Fear and Reason; InSpectres: Library Card Gym Card, Credit Card, Bank; HeroQuest: Hero Bands.

Shared Power: Donjon, Universalis, InSpectres, Great Ork Gods – the gods themselves.

Skill Defaults: If a skill is not possessed by a character, then it defaults to another gauge - GURPS, The Riddle of Steel, Shadowrun.

Skill Grammar: Combine various skills to arrive at a derived skill rank - Ars Magica spell rank system.

Skill Package: lists of skills rising at similar level – Rolemaster Skill Category Rank, Hero System 5th Edition – skill levels can apply to groups of skills.

Solitary-Die Rolls: Roll a single die for contests (as opposed to Dice Pools) -- D&D, Rolemaster, HeroQuest.

Steamroller: a contest success result that has the effect of making it more likely that future contests will succeed.

Story Reward: a reward going to a player for introducing story elements. Capes converts "Debt" into "Story Tokens" in conflicts. Some of these story tokens go to the player who introduced the conflict.

Structured Sessions: Donjon, InSpectres, Paranoia xp.

Tattle: players point out mistakes of other players – Elfs has players tattle on each other when they don't properly play their Idioms.

Timer: Gauge value changes based on passage of time – The Riddle of Steel "Blood Loss," Rolemaster Bleeding, Nobilis natural healing.

Troupe: Having multiple characters so that you can play in more situations. GM role rotates from player to player on a story-by-story basis – Ars Magica.

Turnstile: Force players to make an irrevocable decision when designing some game tool, such as a character. Choosing Race in D&D, Priority Grids in The Riddle of of Steel and Shadowrun.

Unopposed Rolls: D&D, Warhammer.

Verb/Noun pairs: Ars Magica magic system, GURPS magic system.

Win/Lose Contest: a contest where players generate simple success or failure results for character actions. Note that some the following systems have both Degree of Success Contests and Win/Lose Contests: Ars Magica, Call of Cthulhu, Dungeons & Dragons, Elfs, Fudge, Great Ork Gods, GURPS, HARP, HeroQuest, Hero System 5th Edition, Nobilis, The Pool, RIFTS, TORG, Warhammer.

Finis