RFC - Character

Metacosm Development Team

Revision History

Revision 0.2May, 11 2000 Revised by: Entropizer
"The late release" now based on Metacosm-DTD, added section on Character creation
Revision 0.1February, 21th 2000Revised by: Entropizer
"Presidents' Day release" First version based on Entity-RFC (Entropizer)

This document is meant to define what a Character is and describe more precisely concepts that only make sense for Characters such as Skills, memory or roles...

1. License

2. Metacosm

For more information on Metacosm, check our site: Metacosm (http://metacosm.org).

3. Foreword

This document relies heavily on the notion of Entity. To be able to understand all of it, the reader should be familiar with Entity-RFC. Several references are made to that document in the present one. It can therefore be useful to keep a copy of it at hand. Moreover, the reader should keep in mind that this is more an analysis level document and doesn't therefore go deep into the details of implementation. Enjoy!

4. What is a Character?

Character

A Character is a particular kind of Entity. We consider as Character any Entity that Players can control. This implies that a Character is an Entity which is *permanently* controlled by a Controller (whether the Controller is a Player or an Artificial Intelligence). The permanent aspect of the control is important because it excludes all entities that can be *temporarily* controlled because they are under the influence of a spell. However, this is not enough to define a Character since we can imagine that some entities can be permanently controlled by a Controller without being eligible to be a Character (think of machines as an example, we can imagine that their behavior can be controlled by a Controller).

The major quality that an Entity needs to be considered as a Character is (relative) indeterminism. What does this mean is that Players can only play entities which behavior they are able to control at least a minimum. The Player add the indeterminism to the Entity. If a Player doesn't have the possibility of influencing the behavior of an Entity then this Entity is not suited to be controlled by a player and cannot therefore be considered as a Character.

More precisely, it is an Entity which Type (see Entity-RFC) is Character. Characters are also likely to belong to more specific sub-types. Once again, as mentioned in Entity-RFC, on the implementation side, Types are likely to be implemented as classes.

However, in the case of Character, it is more likely that this Type will be implemented as an interface so that Entities from different sub-types could become eligible to be Characters. This document will not however go into the details of sub-types since we consider they are part of a particular instance of the Metacosm game engine. However, we will try to keep in mind the fact that it should be possible to define sub-types without having to actually code new ones. Therefore, we will try to define Characters so as to comply to this requirement.

5. Roles

Highly experimental section... The Role concept is not entirely clear.

Role

Roles are a new concept (?) that could bring an interesting dimension to the game. As their name implies, Roles are roles that a Character can play. A Character can play multiple roles at the same time. It can also change roles or "learn" new roles.

Roles are closely linked to Actions. Actions can define a list of Roles that are needed for its completion with fluency levels. In this case, Roles gather all the needed information that allows a Character to execute a given action such as needed Skills and requirements on attributes. Similarly, Roles can define a list of Actions that they enable Characters to do.

A Character has a ’fluency’ level for each roles that it plays. These levels determine how easy it is for this Character to accomplish the related Action.

Roles can be instantaneous or semi-permanent. They can be as large-grained and semi-permanent as a Character's Orientations (see below) or small-grained and instantaneous as "instant roles" that are played for a very small lapse of time by Characters and that act mainly as a facilitator for interactions. Instantaneous Roles will be developed in greater details with Actions in the Interactions-RFC.

6. Characters and Characterization Levels (CLs)

In this section, we examine more precisely what CLs mean for Characters and introduce new CLs when appropriate. More precisely, we discuss the impact of a Character's Choice of Life on its diverse Characteristics.

6.1. Character creation

Creating a Character that appeals to the Player is an important step towards having a good time while playing a Metacosm-based game. It is really important that a Player feels comfortable with his Character since it will be his or her avatar in the realm of the game. Being able to play a Character for which you chose its qualities and weaknesses, its attitude towards what happens to it, is really rewarding and is an important element that can bring Players to role-play. Metacosm offers therefore a large choice of customization options.

Choice of life

When a new Character is created, orientations are chosen that provide a frame for the Character's acts. It is what we call the Choice of Life. The Character's acts are evaluated considering its Choice of Life. The Metacosm game engine bases its calculation on the Choice of Life to determine a Character's Karma.

A Character’s Choice of Life contains several aspects of a Character’s personality. When a Player creates a new Character, she is asked to provide information on the Character she wants to play.

Metacosm will provide templates that will be available for modifications by Players to speed up the creation process.

Proposed templates (among others...):

- warrior
- merchant
- scientist
- magician

6.1.1. Orientations

Orientation

> Orientations define the Character’s interests. They try to define the global behavior of the Character and influence the way it will probably evolve.

There are several different Orientations that a Player can choose for its Character. They all act at different level and can be combined to add depth to a Character.

6.1.1.1. Attribute Orientation

Attribute Orientation allows the Character to give a subjective importance to Attributes. This system will try to model the way a Character uses its Attributes to influence the way it will handle situations (for computer-controlled Characters) and determine how a Character’s Attributes are going to evolve.

It should be possible to change Attribute Orientation during the game. However, such changes should be rare and subject to a relaxation time i.e. the change will take a certain amount of time depending on how important the change is. For example, a warrior who decided to give the greatest importance to Power at first since his combat philosophy was oriented towards pure brutality will surely have difficulty to embrace a more subtle approach to martial arts by choosing to give a greater importance to Coordination.

The system will work by attributing a given importance to each Attribute. We can imagine that the system could take two different approaches. First, a point-based system in which each Attribute will receive a given number of points, the total of available points being constant so that giving points to an Attribute will mean having less points available for the others. The other alternative is more simple and doesn't allow for several Attributes to have the same importance as it is the case in the point-based approach. This other system consists in giving a strict order to Attributes that are then ranked from the most important to the less important.

It is worth noting that the Attribute Orientation system should influence the Character generation process in the sense that an Attribute with greater importance will be more likely to have a higher level at creation than an Attribute with a smaller importance. However, this is not a necessity and the Attribute Orientation is not required to be an image of a Character's Attribute levels, i.e. a Character with a great subjective importance on Power doesn't necessarily have its Power Attribute greater than the others.

6.1.1.2. Behavorial Orientation

Behavorial Orientations try to capture the way that a Character is likely to behave like. We will try to identify here only orientations that can be computer-assessed and are susceptible to be implemented as part of a computer-controlled Character's Artificial Intelligence.

Ideas for Behavorial Orientations:

- violent / peaceful will use violence or try to argue?
- curious /
- generous / greedy

- talkative / reserved
- ...

6.1.2. Edges and Flaws

Metacosm's Edges and Flaws system allows Characters to be even more characterized by adding little, polishing touches to the overall description of a Character.

6.1.2.1. Definitions

Edge

> An Edge is a particular kind of Characterization Level that gives an advantage to a Character. This kind of advantage can be of several different types such as bonuses to Characteristics, to Skills or Actions...

Flaw

> A Flaw is a particular kind of Characterization Level that gives a disadvantage to a Character.This kind of disadvantage can be of several different types such as maluses to Characteristics, to Skills or Actions...

6.1.2.2. Formal Description

Can affect attributes, Skills, actions... => explain

6.2. Choice of Life and Roles

The role that an Entity assumes affects the way that it reacts to its environment. As an example, working hard physically would probably improve a creature's stamina. Moreover, a creature's Role leads it to practice certain Skills and therefore improve them but can also leads it to forget about others. A particular Role is the Entity's Orientation.

6.3. Modifiers

Done by CPs... => explain

7. Skills

Skills are more complex than attributes or capabilities. Contrary to how it is for attributes, not all entities possess Skills. Skills are limited to sentient entities (or entities made sentient through different means, be it magic or technology...).

A Character possesses an undefined number of Skills. Those Skills are define at the Type level. They represent basic Skills of an Entity of a given Type. It is possible for a Character to learn new Skills or develop old ones. Characters can however only learn a certain number of Skills at full efficiency. This number is given by a function of his/her/its memory attribute. Full efficiency of a Skill means that the Character is able to potentially use the Skill at its best degree of performance. If the Character tries to learn more Skills, more ancient ones will become "faded". They won't be in immediate memory anymore and the Character won't be able to use them with maximum efficiency. However, after a couple of tries and/or practice, the Character may regain full efficiency in a Skill. We don't think that one can really forget a physical Skill. One might however forget unused knowledge.

Each Skill should be based on an attribute. The Skill will work as a modifier to the attribute to enhance the performance of the Entity. We can also think that if an Entity doesn’t have the necessary Skill to perform a given action, it can optionally default to the appropriate attribute. However, not all actions can use the default attribute or if they do, they should do so with an important penalty as it is the case for knowledge Skills.

Example 1. Mathematics

Example: if you haven’t learn mathematics, it will be very difficult for you to solve a given problem. However, a very clever person could devise a way of solving the problem without the necessary knowledge. This is however highly unlikely and difficult to do. We’ll explain more precisely in another document how Skills will work, especially regarding the use of attributes.

8. Memory

Memory is an important concept to the game. It will allow more realism since each Entity will be allowed to keep track of its deeds and history. Moreover, it should simplify interactions by introducing concepts such as short-term memory. Characterization levels have a great influence on memory. Each CL will introduce its own memory that will be shared by all entities affected by a given CL. CLs will act like a pool of concepts, modifiers and memories that will be shared by all member entities. Sharing memory with a community allows to simplify Entity design by factoring out commonality. Moreover, it will allow the possibility of making the world evolve by operating on CL memories. When a shared memory is changed, all entities sharing it benefit from the change. Thus, we can represent knowledge evolution, rumors, etc. by several kinds of processing on Characterization Pools. A particular short-term memory will be used to store an Entity’s context to allow it to act depending on its context. The most obvious application of this is for basic interactions since this short-term memory will allow an Entity to directly act on objects in its context without having to have to identify each other Entity before acting on it.

Examples of CPs memories:

Example 2. General level

We can imagine that each Entity will have a specific set of actions that will model physical law of the world. This set of physical will manage how entities interact (unconsciously) with their environment. As an example, if an Entity is pushed by another, the action representing the related physical law will be used. For more information, see Interactions-RFC.

Example 3. Type level

"Races" memory could simulate cultural knowledge or specific abilities.

Example 4. Geographic level

Memory of home town history.

Example 5. Community level

Knowledge of rumors, name of the local baker...

Example 6. Professional level

Specific Skills given by the Entity's profession if any.

Example 7. Personal level

Personal Skills, history, discoveries, knowledge, relationships with other entities (friends, enemies...)

It should be possible for a Character to explicitly query its own memory. => explain

9. Possessions

10. Editor