RFC on perceptions

Metacosm Development Team

Revision History

Revision M1 July, 15th 2001
Conversion to Docbook. Definitions for Medium, PerceptionState, Stimulus, Stimuli, Sense, StimuliDispatcher and LowLevelDescription. Description paragraph from Entity-RFC. Explannation on where Stimuli are modified.
Revision 0.6.2January, 7th 2001 Revised by: Janselmeer
Minor changes in 'How To', 'Implementation' comes back with descriptions (LLDs)
Revision 0.5 April, 16th 2000 Revised by: Janselmeer
Changes in 'dazzle section'
Revision 0.4 April, 9th 2000 Revised by: Janselmeer
Adds in 'senses' section, new 'relay/converter' section, adds in 'focus' section
Revision 0.3 March, 20 2000 Revised by: Janselmeer
Adds in 'descriptions' section. Creation of 'How to' section. Conversion into metacosm-DTD
Revision 0.2 February, 23 2000Revised by: Janselmeer
'Implementation' section removed and replaced by 'StimuliDispatcher' section. Addition of a paragraph on sensitive memory in 'Sense' section. Addition in 'Reflection' section
Revision 0.1 January, 4th 2000 Revised by: Janselmeer
First version gathering ideas from utopie mailing list and the first document drafts

This document specifies the perceptions system.

1. License

MA 02111-1307, USA.

2. Metacosm

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

3. Notations

Some terms are defined in the glossary, available on the official site.

4. Stimuli

The perceptions system is entirely based on Stimuli emission by Entities and on receiving of these Stimuli by other Entities.

There are two kinds of Stimuli. We can distinguish:

- Description (or state) Stimuli which describe the state of the associated Entity. These Stimuli can last an infinite time.

  Example 1. Description Stimuli duration

  The torch is giving out light: 6 hours duration.

  The flag is red: infinite duration.

- Action Stimuli describe actions of associated Entities. Their duration is instantaneous.

Example 2. Action Stimuli

Janselmeer lights the torch.

The torch goes out.

The dragon utters a deafening growl.

Medium

a means of effecting or conveying something (Merriam-Webster's Collegiate Dictionary)

channel for the transmission of some Stimuli: only Stimuli corresponding to the medium can be transmitted by this medium.

Stimuli can also have multiple shapes and intensities. A Stimulus can thus go down the following Media:

- Visible light
- Infrared
- Ultraviolet
- Sound
- Infra-sound
- Ultrasound
- Mechanic (touch)
- Flavour (taste)
- Smell
- Magnetic field

- Electric field
- Life
- Magic
- Invisible
- Telepathy
- Feeling (to feel real feelings of somebody)
- ...

Its intensity is given by a positive real number. A value of 1 means a normal (or medium) intensity. Below 1, intensity is low. Above 1, intensity is high.

A scale can be built by applying a decimal logarithm:

```
<0.001 -> <-3: exceptionally weak intensity.
0.001  -> -3: very weak intensity.
0.01   -> -2: weak intensity.
0.1    -> -1: low intensity.
1      -> 0: medium intensity.
10     -> 1: high intensity.
100    -> 2: strong intensity.
1000   -> 3: very strong intensity.
>1000  -> >3: exceptionally strong intensity.
```

5. Senses

In order to be able to receive Stimuli, Entities possess Senses. Each Sense is specialised for Stimuli reception in a determined Medium. For example, a normal human has 5

Senses:

- Sight -> light
- Hearing -> sound
- Touch -> mechanical stimulus
- Taste -> food
- Smell -> smell

Each Sense has also an intrinsic power. The more powerful a Sense, the more it will be able to receive weak Stimuli.

As a Stimulus intensity, a Sense capacity is given by positive real values. A logarithmic scale identical to the previous one for Stimuli can be built:

```
<0.001 -> <-3: exceptionally weak power.
0.001  -> -3: very weak power.
0.01   -> -2: weak power.
0.1    -> -1: low power.
1      -> 0: medium power.
10     -> 1: high power.
100    -> 2: strong power.
1000   -> 3: very strong power.
>1000  -> >3: exceptionally strong power.
```

In order to know if a Stimulus has been received by a Sense, we have to multiply the two powers (or add the two values on the logarithmic scale). With a value of 1, the Stimulus is received normally. Below 1, the Stimulus is badly received. Above 1, the Stimulus is very well received.

```
Stimulus | Sense | reception
----------------------------
   1        1        1       -> received
```

```
 1       0.1      0.1      -> not well received
 1       10       10       -> well received
0.1       1       0.1      -> not well received
0.1      0.1      0.01     -> not well received
0.1      10       1        -> received
10        1       10       -> well received
10       0.1      1        -> received
10       10       100      -> very well received
```

According to the reception level, the Stimulus description can vary.

As soon as it is received, the Stimulus is temporarily kept in a sensitive memory. So the Entity can consult received Stimuli at any time without having to ask their reemission to the source for all that. This memory is completely independent of the Entity memory and is managed in a totally autonomous way.

```
 ----------- explicit obs.  ------------        ----------
 |  New    |---------------->| Sensitive|<------>| Entity |
 | Stimuli |---------------->|  memory  |        |        |
 ----------- high intensity ------------        ----------
             Stimuli (*)

(*) level = fct(concentration) ;
```

It works in fact in the same way as a cache memory, by keeping the last received Stimuli or the most consulted ones whereas the other ones will be erased (or forgotten) when it will be needed to free space.

Anyway, Stimuli will stay a limited time in this memory. It is indeed rather natural to not feel a Stimulus anymore after some time when it does not evolve. For instance, when one encounters an unusual smell, one notices it at once and one ends up not paying attention anymore some minutes later...

Stimuli are filtered by Senses and transformed into descriptions (see Descriptions). Controllers can access only to these descriptions. They get them directly through

Senses and not from the Entity's Memory (there is already the PerceptionState as a sensitive memory).

6. Focus

In normal times, an Entity receives all Stimuli equally. However, it is able by concentrating to focus on particular Stimuli. Their perception by the Entity is then increased, whereas the perception of other Stimuli is decreased. The focus capability depends on the concentration of the Entity. Some skills can improve it too. For example, musician can focus better on sounds because of training.

It is also possible to focus:

- Either on all Stimuli coming from a particular Entity,
- Either on all Stimuli of a given Medium.

A Medium (or Sense) focus is improved when the number of active Senses is reduced. For example, it is easier to hear something when closing eyes.

The 'focus' command allows to realize this by the following means:

- focus <sense> -> the Entity focuses on Stimuli associated to <sense>
- focus <entity> -> the Entity focuses on Stimuli coming from <entity>
- focus -> the Entity switches back to a normal perception.

7. Dazzle phenomenon

Dazzle phenomenon occurs when a very strong Stimulus hides weaker Stimuli. For example, the sun hides the stars. The system proposed in this section should allow to simulate this phenomenon. To avoid all misunderstandings, we insist on the fact that it is no matter here of Sense adaptation (for instance, you are in the dark, somebody light on, you are dazzled and you cannot see anything, then your eyes adapt to the light and you can see again). As a simplification, any possible adaptation will be considered as instantaneous.

Our model is a question of punctually decreasing a Sense power according to the power of the most powerful received Stimulus. This Stimulus must have a positive duration, that is it is not instantaneous.

The rule is the following:

Effective Sense power = intrinsic Sense power / (intensity of strongest received Stimulus ** Sense control)

or

Effective Sense power = intrinsic Sense power - (intensity of strongest received Stimulus * Sense control) when getting values on logarithmic scale.

In all events the most intense received Stimulus should have an intensity above average (1 or 0 on logarithmic scale). Otherwise the power of other Senses would be increased and not decreased. The 'sense control' parameter is an attribute of the Sense. It represents the ability of the Sense to resist to dazzle. It is a positive float with a default value of 1.0. A Sense with a control lesser than 1.0 resists well to dazzle. A sense with a control greater than 1.0 resists badly to dazzle.

It is worth noting that the most powerful Senses are less easily dazzled than weaker Senses if their control are equal.

Example 3. Reception with dazzle phenomenon

```
      Dazzling | Received |       |         |
      Stimulus | Stimulus | Sense | control | reception
      ----------------------------------------------
         10         1         1       1        0.1  -
  > not well received
         10         1        10       1        1    -> received
         10         1       100       1        10   -> well received
         10         1       100       2        1    -> received
```

```
        100        1       100       2        0.01 -
> not well received
        100        1         1       1/2      1    -> received
```

8. Reflection

This section is based on the physical phenomenon called the same. Here is a simple example concerning light: let us imagine a white coloured object (that is it reflects light on all light spectrum) of any shape (a cube for instance). If we plunge this object into darkness, we won't see anything: it does not emit light. If we light a red light, it takes the red colour. If we light a blue light, it becomes blue and so on. It is in fact a matter of reflection: the object sends back a light of the received colour.

Entities in game will be able altogether to reflect received Stimuli in the same way. Indeed each received Stimulus can influence the state of some entities and modify their Stimuli emission. However, we must pay attention to the fact that an Entity which receives blue light and yellow light will not send back two Stimuli of the two colours but only a green one. In fact each Stimulus has modified the Entity state and the Entity emits new Stimuli from this modified state.

9. Relay/converter

An Entity receives a Stimulus. Let us guess that it sends back a Stimulus of a different type. Let us take the phone example. We can consider that it is an Entity which is able to receive or emit electromagnetic or acoustic stimuli. Nevertheless, if it is thrust into an isolated system, it will not do anything. In fact, it does nothing more than turning sound waves into electromagnetic waves and vice versa. This is a Stimuli converter.

We can also imagine an Entity of a special type made of several parts split up between different locations. This stays hypothetical, but this could give a remote transmitter-receiver able to receive Stimuli in a Place and to send them back in another.

Perspectives given by such Entities are very interesting, as Stimuli relays from a container to another (refer to "StimuliDispatcher" section).

10. Descriptions

An Entity is a construct with which other Entities and game constructs can interact. A successful interaction can only be the result of an "informed decision". It is therefore necessary to establish a way to obtain easily processable information about an Entity.

At a higher level, even if all interaction is done through game constructs, players will need to learn more about the world their Character will evolve into. It is therefore necessary to provide a way to describe Entities in an intelligible way.

Another issue is the need for more realism compared to older MUDs. Typically, descriptions were stored in text files, one (or sometimes several) description(s) per Creature type. This had several drawbacks. First, each Creature of a given type had exactly the same description as any other of the same type. Second, the description was static, i.e. it did not evolve at the same time as the Creature it was representing (no monitoring of health condition, growth, change ot clothes...).

An Entity receives the whole information about its environment thanks to Stimuli. For instance, on receiving a visual Stimulus from a Creature, the Entity could get its identity, its size, its race, its rough age, its rough weight... A sound Stimulus could also give the Entity some other information. This raw data is sufficient for a bot but for a human player, it must be laid out in a textual, graphical, sound or some other way... This is the objective of the dynamic description mechanism. We'll now show how it works for textual descriptions, but it can be adapt in most cases to other descriptions types.

Example 4. An Entity with some attributes and the descriptions which could be generated

```
identity: EntityID
race: wolf
size: 2m
```

```
colour: grey
position: x y z (3D coordinates)
```

The description generated on the fly is: "A huge grey wolf is standing a few meters in front of you.". If the wolf was already known as "Medor" by the Character (that is if the EntityID Entity was in the character memory), the description could become: "Medor is standing a few meters in front of you.".

A Stimulus carries therefore a certain amount of information which can be used afterwards by the Entities which receive it. However, we have seen in previous sections that a Stimulus could be more or less well received. So this information will not always be accessible and could also be distorted. Some information can require first-rate perception too whereas another will be easily obtained.

Example 5. Example with an action

```
Subject Entity: EntityID (Raoul)
Action: ActionID (eat)
Object Entity: EntityID (bread)
...
```

The description will be: "Raoul is eating some bread."

If a Stimulus is badly received, this description can be replaced by one of the following:

```
"Somebody is eating some bread."
"Raoul is doing something with bread."
"Raoul is eating something."
"Somebody is doing something with something."
...
```

Or the description could even be completely wrong:

```
"Raoul is threatening you with bread."
```

```
"Raoul is eating some wood."
```

The problem which appears is how to know which part(s) of descriptions are altered. The problem exists not only for players but for bots too, because it concerns attributes at the source. We associate a minimal reception index to each attribute. The Stimulus reception level should be greater than this index value to have an optimal reading of the attribute.

To better understand, let’s continue with the previous example:

Example 6. Example with an action (2)

```
Subject Entity: EntityID (Raoul): 0.01
Action: ActionID (eat): 7
Object Entity: EntityID (bread): 1

Reception
10: "Raoul is eating some bread."
1: "Raoul is eating something."
0.1: "Raoul is threatening you with a bread".
0.01: "Raoul is eating some wood."
```

but, contrary to this example, it won’t be as simple to practically give a wrong value to attributes, except to numerical values (size, length, weight...). We could in theses cases determine the wrong value in an interval centred on the real value and with a diameter increasing when reception level decreases.

Now see an other subject. In these examples information is reduced and descriptions are therefore short. Nevertheless, with state Stimuli, descriptions (at least textual ones) may well be too long. As a consequence, a detailed (long) description and a shorter one have to be generated. The short description is the default one but the user can consult the long one at leisure. This can apply as well to texts or other description types. For example, we can imagine a graphical interface where an Entity appear as a picture (short description) and clicking on it displays a window showing additional information on the Entity.

Example 7. Example with an action (3)

short description: "Raoul is standing behind his counter."

long description: "Raoul is standing behind his counter. He is a stout man of medium height. He is wearing a 'marcel' and white trousers held up by a pair of braces."

11. Implementation

11.1. StimuliDispatcher

A StimuliDispatcher is an intermediary between Entities emitting Stimuli and Entities receiving them. There is a StimuliDispatcher associated to each Place.

The operation is the following: when an Entity come into a new Place, it is registered to the local StimuliDispatcher as a new Entity able to send Stimuli (we consider that any Entity can potentially emit Stimuli even if it could be not always the case), and able to receive them with each of its Senses. The StimuliDispatcher asks then to all registered Entities for emitting Stimuli again in order for the new Entity to have a glimpse of its environment. Afterwards, when a Stimulus is emitted by any Entity, the StimuliDispatcher dispatches it to all registered Senses which can receive it. So the StimuliDispatcher role is limited to Stimuli delivery only; on no account does it have to modify Stimuli.

11.2. PerceptionState

PerceptionState

> The PerceptionState is a cache containing state Stimuli emitted by an Entity (one Stimulus per Medium). Each Entity should have a PerceptionState.

The PerceptionState prevents ever recreating the same Stimuli for and Entity before they are sent to any StimuliDispatcher. Assuming the content of the Stimuli will not change quite often, the benefits should be important.

11.3. Descriptions

Before going any further in this section, the Entity-RFC should be well known by you. If you do not know what an Entity, a Property, a Type or an Influence, are please read the Entity-RFC.

In earlier sections, we have introduced Stimuli. Here we deal with how to fill them. To do that, we define a set of description Properties besides the usual Properties which we may qualify state (or internal) Properties. The description Properties are calculated from the state Properties.

Example 8. Raoul the baker

```
State Properties:
-----------------
identity: EntityID
race: human
name: Raoul
job: baker

Description Properties:
----------------------
identity: EntityID
race: human
perfume: odor of bread.
```

In this example, there are 3 descriptions Properties. 'identity' is necessary because it allows Bots to identify other Entities. Then there is 'race' that is equal to the state Property of the same name, and 'perfume' which was created from the state Property 'job'.

Now we must sort the description Properties by Medium because every description property cannot be perceived through all Media. In the same time, we associate an intensity to each of them. The result is a table. We may call it the Low Level Description because it contains any necessary information to build a description (text, sound, ...) of the Entity.

Example 9. Raoul the baker (2)

```
              View                        |            Smell
        -------------------------------------------------------------
  ------
        identity: EntityId      1         |  identity: Enti-
  tyID              1
        race:     human         2         |  race:     hu-
  man                 1
                                          |  per-
  fume:  odor of bread        5
```

Every Entity will have such a table. Entity editors (or zone makers) must of course provide how to build it in the Entity Type and Influences.

The last step is now to create the Stimuli that will carry data through Media. It is quite simple: we just have to put the content of each column in a Stimulus related to the same Medium than the column. The Stimulus will take place in the PerceptionState. The only issue is if we should copy the column by value or by reference? The only flaw that can happen with references is when the description of an Entity is modified after the Stimulus is send. When the Stimulus will be received, it will contain the new value and not the old one. But this is not really a problem. On the other hand references avoid the creation of lots of Java objects and this is a rather great advantage. That is why we decided that Stimuli will contain references on a column.

12. How to

This section intends to show how the perceptions system defined in previous sections can be used in unusual cases. It concerns cases which can't appear in the real world and which are created particularly be magical effects.

- How to do an ambient Stimulus?

  You only have to put the Place as Stimulus source.

- How to do an unlocated Stimulus?

  For example, we would emit a Stimulus from the town A when being in the town B. The sender Entity only have to register to the StimuliDispatcher linked to B, to send its Stimulus and to unregister.

- How to ventriloquate?

  You only have to send a sound Stimulus specifying that source is the receiver and not the real sender.

- How to do a worldwide Stimulus?

  You only have to relay a ambient Stimulus to all StimuliDispatcher.

- How to modify by illusion the appearance of an Entity?

  For example, how to give a gold bar appearance to a lead bar or to give a mouse appearance to an elephant.

  For it, the target Entity should emit a Stimulus which doesn't correspond to its real internal state (the lead bar has still the same weight and not the gold bar one). For instance, assume a lead bar emits the following visual stimulus:

  ```
  identity Entity: EntityID (Raoul)
  color: grey
  ```

  Cast an illusion spell and the lead bar will look like:

```
identity Entity: EntityID (Raoul)
color: gold
```

13. Definitions and summary

Stimulus

- a source Entity
- an intrinsic intensity
- a Medium
- an action if need be
- some attributes to describe the source or the action

Stimuli

Stimuli = Action Stimuli + Description Stimuli

Sense

- a medium
- a sensibility (power)
- a dazzle control
- a memory of the last received Stimuli

StimuliDispatcher

A StimuliDispatcher is an intermediary between Entities emitting Stimuli and Entities receiving them. There is a StimuliDispatcher associated to each Place.

- a set of source Entities.
- a set of receiving Senses per different Medium.

LowLevelDescription

Table containing all the necessary info to build a description of an Entity (description properties with intensity, for each Medium)

New commands:

- focus: allows to focus on particular stimuli