# Chapter 1

# Diage Modelling Language

To help me visualise the stories I wanted to tell within Rouge, I needed a tool in which I could quickly draw up a narrative scenario to test different operators that the Narrator would perform. For this purpose I developed the Diage Modeling Language, or DML. DML is a very simple modeling language akin to those found in the UML specification. With DML I could create a scenario and declare certain operators to create a finite amount of possible story outcomes. It served it's usage as a prototyping tool and set the benchmark for what would become the Narrator.

Some functionalities that Rouge has were not conceived yet during the development of DML. The most noticeable are the actor attributes. That system was brought into the Narrator after the half way point of research and development time, in which DML was not actively used any more.

DML served its purpose for the early development phase by allowing me a quick and easy way to set up story prototypes, define the working operators that a narrative planner could use, and discover what kind of variations a - then hypothetical - planner could come up with.

This chapter serves as a run down of the DML specification, listing the applications and possible future work relating to it.

## 1.1 The structure of DML

DML is structured with the use of entities. These entities in four forms; `Actors`, `Objects`, and `Spaces`. The latter is also an information container which I will discuss in section 5.2. The following section will only cover the pure entities; the objects and actors. In conjunction with `actions` and `accessors` these entities convey story information and plot progression.

### 1.1.1 Objects

Rouge objects form the basis of all entities. They represent the props and items that we find in the world that have narrative importance. For example; if the Player walks in to a shop Rouge does not specify all items that one could buy in the shop, but only those that have plot importance. In terms, these objects should adhere to the *Checkhov's Gun* principle. This dramatic principle states that all objects used in a narrative should eventually be used. I quote: "*One must never place a loaded rifle on the stage if it isn't going to go off. It's wrong to make promises you don't mean to keep.*[1]" Objects have three properties; a `name`, a `ID` and a `type` The ID is a unique identifier dependant on the type. And the type is used with actions/accessors as seen in figure 5.2 (Futher discussed in section 5.1.2). The name is the noun given to an object within the story context. For example; The player receives the *Skeleton Key of Awesomeness*, but the type is just `key` giving it no special properties than any other key. It might make sense to name an object something else than it's type, but a name is not given it defaults to it's type. Objects form the world, and all other entities derive from the Rouge object. This means that all entities have the same properties as the object, but can extend upon it.

[1] `http://berlin.wolf.ox.ac.uk/lists/quotations/quotations_by_ib.html`

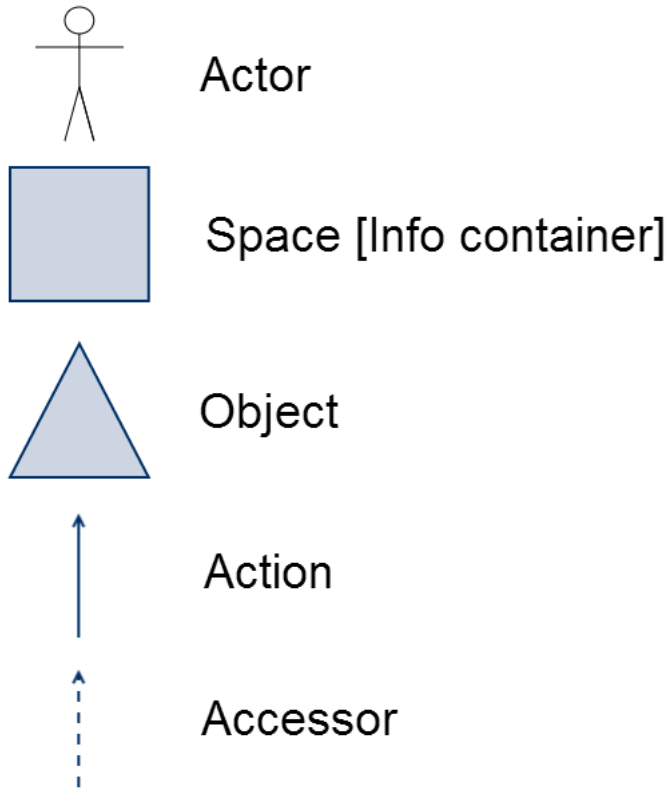



Figure 1.1: DML Symbols

### 1.1.2 Actors

An actor is the representation of any one object that can, as the noun implies, act. Examples are the store-clerk, a wandering adventurer or the player. The actor is the only entity that can physically interact with the world, and by doing so the only that can change the world's state. By being able to change the world, the actors are the only entities that can ensure plot progression. Just like the object, an actor has three properties; a `name`, a `ID` and a `type`. The type property is used in predefined actions as seen in figure 5.2a. This figure defines that the `Player` can **speak** to all actors of type `NPC`. A further glance at figure 5.2 shows some more actions that could be defined for the player actor. These predefined actions tell us that the player can trigger all events and enter all spaces. I will expand upon these actions in section 5.3.

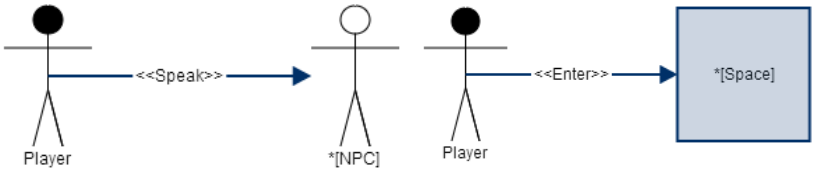
## 1.2 Spaces

Information containers are entities that hold story information. A space holds information about it's spacial children Information containers have the unique property that they are nestable. For example; a space that represents a city can hold several spaces that represents housing.

A space is the representation of any segment of the world or the world itself. As spaces are info containers they are nestable, as mentioned before, but they differ in the fact that they can hold every entity as a child. These children make up the spacial awareness of the space and tells us what information it can pas on to actors. Usually an actor gains all the information a space can give upon the moment it enters the space; when the actor becomes a child object to the space. This can be modified, and some parts of information maybe withheld from the actor, but this is where the events come in.

## 1.3 Actions and Accessors

Actions and accessors are the abstract connectors in Rouge. They convey what actors can do (actions) and what knowledge they possess (accessors). Some accessors are implied, due to the fact that an actor might be the child of a space, in other cases these connections are explicitly added to a Rouge model. If we review the diagram in figure 5.3 we see that the mayor has no connections whatsoever. This would imply that the Mayor has no knowledge about what's or who's in the store. If we compare that figure to figure 5.5, we see that the Mayor now has a connection to the Shop, thus we can be sure that the Mayor has the knowledge it would have, if it had been in the shop space itself.




(a) A standard action for NPC interaction (b) A standard action for space interaction

Figure 1.2: Predefined actions

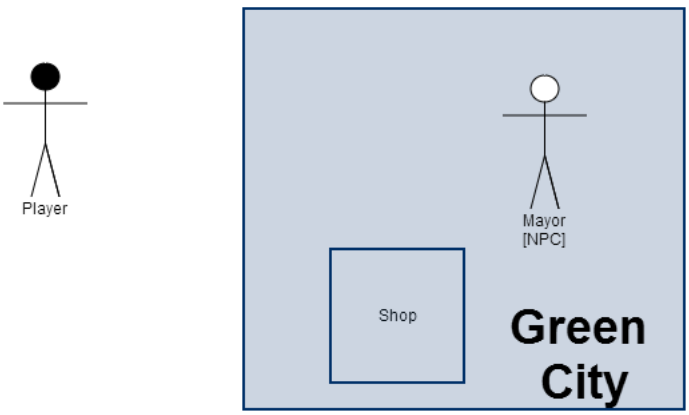



Figure 1.3: An example of a Diage diagram using predefined actions

## 1.4 The narration

A DML diagram is a snapshot of the current narrative setting, and displays all relevant information to that setting. One look at the diagram should tell you what a given entity knows about its surroundings. This information can be conveyed whenever a entity interacts with an other entity.

As a rule, only actors are allowed to record information. Objects are still-lives and as such do not have spacial awareness, but may give information to the actor. The prime example is a treasure map that an actor picked up. The object in question contains the information on a possible treasure as located by the map. This information gets transferred the same way that an actor could learn new information from an other actor.

As a snapshot of the given scene, DML loses the information of the scenario's before and after it. To gain a sense of story progression, one needs to look at several diagrams in quick succession to see what is really happening. Observe the following example:

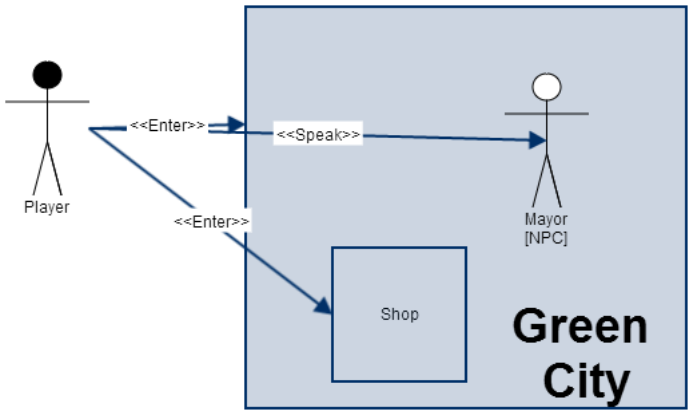



Figure 1.4: An example of a Diage diagram without using predefined actions

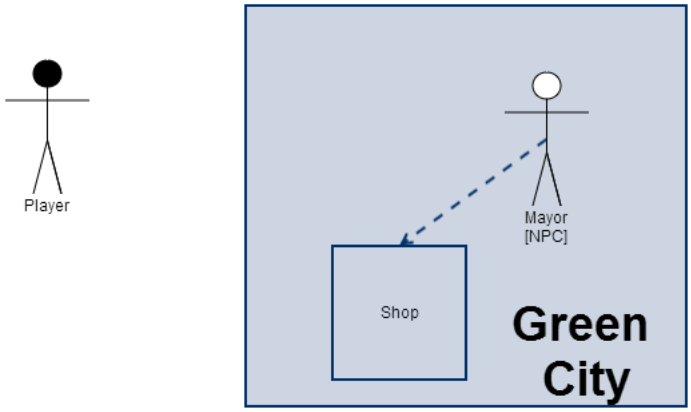
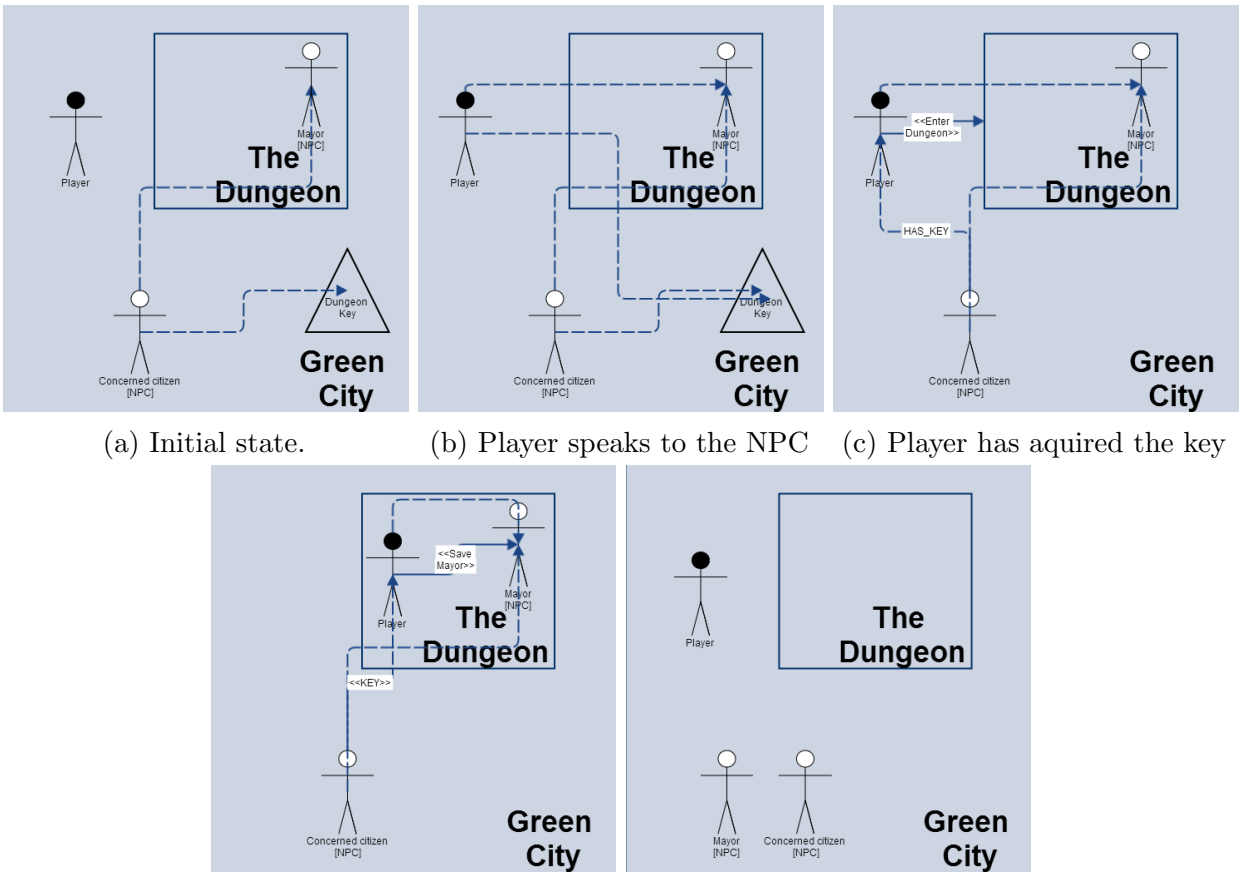



Figure 1.5: A Rouge diagram showing a explicit connection between the Mayor and the Shop




(a) Initial state. (b) Player speaks to the NPC (c) Player has aquired the key

(d) The player entered the dungeon and can now try to save the *Mayor* (e) The mayor is rescued from the dungeon! You stallwart hero of the land!

Figure 1.6: An example of a narrative in DML

These five diagrams tells us what happened within this particular story segment. The mayor is trapped, and the concerned citizen asks the player for help. To get into the dungeon the player needs the dungeon key, and helpfully the citizen told him where he could find that key. After unlocking and entering the dungeon, the player will have found the mayor in some kind of distressful state, and leads him outside. The point is, that we need the knowledge of previous story states, before we can assume that we know what is happening. DML has not been designed as a language that tells a story, but purely a way to display the state of one at a certain point of time. By manipulating the exposed operators of a given diagram, we can see what different paths the planer could have taken at a given scenario.