Diage: A Dialogue Generator

Nico Andrew Glas

November 25, 2013

Abstract

will come later[1] [2] [3] [4] [9] [6] [5] [7] [8]

Acknowledgements

Contents

Chapter 1

Introduction

In the world of game development we strive to create the best experiences for a wide and diverse audience. During the course of development there are a number of obstacles that can (and often will) hinder the progress of said game. Some of these hindrances come from processes that are essential to a game like; level- and world building, or story- and quest design. They take a proportionally large amount of development time and take a very specific mindset to create. A current development within the game development[1] community to tackle these issues is the use of *Procedural Content Generation* (PCG). This may be a confusing phrase for some, so let's dissect it. **Content Generation**: When taken apart, this part of the phrase will probably become a lot clearer. It just really means what it says, we try to generate content, be that a quest for a Role-Playing game (like *World of Warcraft*) or a complete dungeon for a dungeon crawler (like *Rogue* or the *Diablo* series). This generation can take place either on run-time (also called `live` or `on-line` generation), making it possible for new generation for each time the game is run. Another possibility is to generate content prior to release. This has the benefit of being able to tweak the content to make it more satisfying or aesthetically pleasing, but removes the dynamic aspect that some games might strive for. Nobody will say that one of these methods is better than the other, but they are both tools to be used on the right moment. **Procedural**: This part of the phrase is the one that often causes the most confusion, but just means that we use a specific procedure to generate our content. In the world of programming, it usually comes down to one or multiple algorithms that work together to generate predictable content. A good example for a non-digital form of procedurally generated content is the card game *Klondike* (also called *Solitaire* of *Patience* in the US and the UK respectively). The procedure in this game is the shuffling of the deck. This "procedure" ensures a random order of the cards, and the resulting content is the layout of the game. While this might be a good example for content generation it is a poorly implemented one, for there exists multiple outcomes that the game can not be completed.

In my thesis I strive to discover how a *roguelike* game can benefit from a procedurally generated narrative.

1.1 On "random" generation

In former years we always spoke of content that was generated randomly. The use of this adjective has since been frowned upon, because random insinuates a lack of control and predictability. We now favour the term procedural, as this covers the use of predictable algorithms and a structure that is mathematically justified. Generally speaking the two phrases are interchangeable, but their sentiment can cause confusion. For the sake of coherency I will continue using the term *procedural content generation*.

[1] twice development within the space of 3 words. NOT DONE

1.2 A word on narration

I want to go as far as to say that narration defines what being humans means. We, as a society and as a species, have always used a storytelling context to receive and deliver information. Be that a story on the dangers of bears and lions to a more modern setting for the sake of leisure. But even then, for the sake of argument we need a proper definition of a narrative. Riedl and Young stated that a narrative is in it's simplest form a temporally ordered sequence of events [7]. This is the definition I will keep to throughout this document. One other thing of particular note; a narrative does not explicitly mean the usage of text or spoken word. This is an easy, and clear way to convey a story, but never the only ways to do so. Granted; it can certainly add to the experience, but in my own opinion a game should, first and foremost, try to convey a story through it's mechanics. To discard that rule is to introduce the expensively named *ludo-narrative dissonance*, or a discord between narrative and gameplay. There has been a vast discussion on the Internet about this phenomenon, on which I will not elaborate, but is worth of note to any aspiring game developer/designer. Games can be a powerful medium in which to explore the human condition, but for that we can really come to that point we need to start treating it with that same respect. And that's my preachy part.. which probably needs to be cut[2].

1.3 A word on static and dynamic generation

Throughout this document I reference to static and dynamic generation, or static and dynamic narratives. These terms refer to the point in time where the generation takes place. When we speak of a static generation, this happens during the development. A developer may choose to generate a game world and continue to populate that world with the rest of his content. In a dynamic setting, said world is generated at runtime, usually when a new game is started. This means that every time the player starts a new game he gets a new world to explore. In a narrative context that means that a static narrative has been generated, but will never change. Whereas a dynamic narrative will always try to be different from the former.

1.4 Create-IT

Create-IT applied research is one of the research institutes hosted by the University of Applied Sciences of Amsterdam. In this lab students, teachers and researchers all preform applied studies in the different sections of the IT world. Their goal is to educate future professionals in the uses of applied research, so these professionals can anticipate the ever changing field that is Information Technology. In the newly created Game Research Lab students and researchers alike contribute to the ever growing community of game developers; making the development of games easier or trying to understand current problems within the industry[3].

1.5 Problem definition

My main research question is *How can a roguelike game benefit from a procedurally generated narrative?*. The other questions I want to answer are: "How does a computer recognise a good narrative", and "How do you structured a procedurally generated narrative". In this section I will state my research questions, and try to dissect them so all readers of this document have the same definitions, and the same context.

How can a roguelike game benefit from a procedurally generated narrative?

[2]Which makes me sad

[3]How much can a few lines suck?

Let me clarify the term *roguelike* . The genre started with the video game *Rogue* that was released in 1980, and was characterised by having "random" dungeons where the player has to navigate rooms and fight monsters. The ultimate goal of the game was to get to the highest level possible without dying once. The game never really "ended". The game was over when the player died, but after that the player got to start all over again on level 1 with a complete newly generated dungeon. As the game gets progressively harder when the player starts go get to other levels, the chances for the player to lose get higher. Now, back to the term "*roguelike* "; A game with no definitive end and *perma-death*[4]. The previous years has seen a rise in popular *roguelike* games, *FTL: Faster Than Light* by **Subset Games** (2012) and *The Binding of Isaac* by **Edmund McMillen and Floris Himsl** (2011) being just some examples.

I specifically target *roguelikes* for their inherent use of content generation. The need to have a different set of content throughout a play-session is the key selling point of a *roguelike* game. This makes it the right genre to experiment in with any new type of content generation.

In my question I speak of benefits to the game by the use of a procedurally generated narrative. This results in a few sub-questions that tackle these benefits. *What gameplay benefits can we introduce*, and *How does the development process benefit from a procedurally generated narrative.* These questions are measurable to a certain degree that give us a good view on the beneficial factor of a generated narrative.

1.6 Research methods

In the previous section I have declared my research questions being

1. *How can a roguelike game benefit from a procedurally generated narrative?*
   (a) *What gameplay benefits can we introduce?*
   (b) *How does the development process benefit from a procedurally generated narrative?*

This section will cover the methods used to measure the benefits that adding a procedurally generated narrative has.

1.6.1 RQ 1a

What gameplay benefits can we introduce? If there ever was a noun which definition was disputed, it's probably *gameplay.* I'm not going to enter the discussion at this point, but just stick to the one I feel covers most facets: *"The experience of gameplay is one of interacting with a game design in the performance of cognitive tasks, with a variety of emotions arising from or associated with different elements of motivation, task performance and completion."*(Lindley, et al. 2008). [something about the quote]. I want the generated narrative to be part of a game, not just a additive thereon. With the close ties generated content has with emergent behaviour, I want to explore the changes that get introduced when adding a dynamic narrative to gameplay.

1.6.2 RQ 1b

How does the development process benefit from a procedurally generated narrative? This questions has some snags. What does it take to measure a process? Do we look at time spent writing a story and contrast that with the time spent building a story generator? What about the fact that we only have to build that generator once, whereas story crafting needs to be done for every single game.

[4]Perma-death is a term used for the loss of progress that a player experiences when his/her character dies

1.6.3 Deliverables

As a deliverable I will create a small *roguelike* game that serves as a proof of concept. With a project duration of 20 weeks, any form of "finished" game is out of scope. The game should have the essentials to be recognised as a *roguelike* , but it's *usp* is a form of generated narrative. The actual outcome is to be decided[5]. A possible deliverable will be a scientific paper with the intent of publication. However, this is speculative and tentative, for at the moment we do not know what results we will generate.

[5]will be included in the final draft

Chapter 2

The interactive story

Here I want to talk about the field of interactive storytelling and narrative building. Gives us a nice entrance to the rest of the document

Chapter 3

Diage Modelling Language

In this chapter I will cover the Diage Modelling Language (DML) that is used to visualize the flow of information, some are static and others will wait for the interaction of the player to release this information and ensuring plot progression. *Diage* uses the symbols to represent the *Diage* entities as seen in figure 3.1.

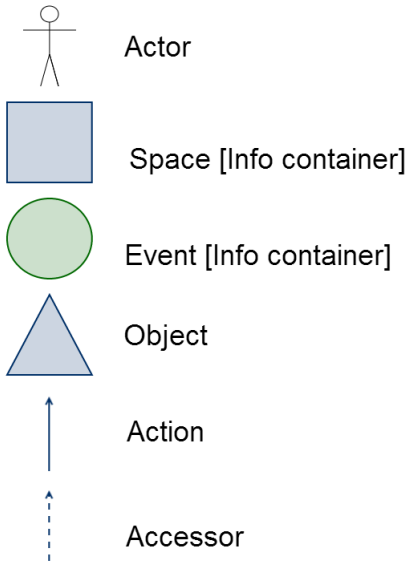


Figure 3.1: DML Symbols

3.1 Entities

Diage entities come in four forms; `Actors`, `Objects`, `Spaces` and `Events`. The latter two are also information containers which I will discuss in section 3.2. 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.

3.1.1 Objects

Diage 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 *Diage* 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 3.2 (Futher discussed in section 3.1.2). 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 *Diage* object. This means that all entities have the same properties as the object, but can extend upon it.

3.1.2 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 3.2a. This figure defines that the `Player` can **speak** to all actors of type `NPC`. A further glance at figure 3.2 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 3.3.

3.2 Information Containers

Information containers are entities that hold story information. A space holds information about it's spacial children, and events release information into spaces when they are resolved. 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.

3.2.1 Spaces

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.

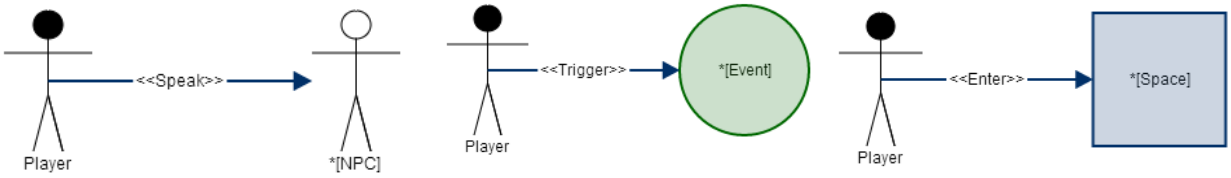
3.2.2 Events

A event is the odd one out as an entity, as it is the only one that does not represent something within the narrative. A event is the abstraction of information that is released into the story when an actor - usually the player - interacts with it, thus events are used for story pacing and narrative convenience. When we need the state of a space to change we use a event to initiate that change. This only applies on the narrative context of the world, because *Diage* does not specify everything that happens with in the interactive context. For example; if the player went to a store to buy some cheese to eat. *Diage* only specifies the store's loss of the cheese, if said food item is a special narrative item. Like a poisonous piece of cheese that the villain left there as a cunning trap for our hero. Events make the world go round and are the dynamic forces in *Diage* .

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

3.3 Actions and Accessors

Actions and accessors are the abstract connectors in *Diage* . 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 *Diage* model. If we review the diagram in figure 3.3 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 3.5, 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 event interaction (c) A standard action for space interaction

Figure 3.2: Predefined actions

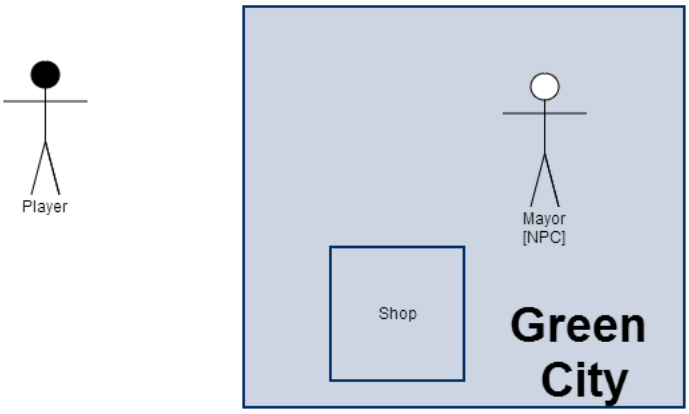


Figure 3.3: An example of a Diage diagram using predefined actions

3.4 Diage as a graph

Diage can be defined by DML, but also as a graph. This form of representation allows us to use graph theory to manipulate and transform the diagram, which we'll cover in the next chapter[2]

[2]SHOULD BE LEFT FOR FINAL!

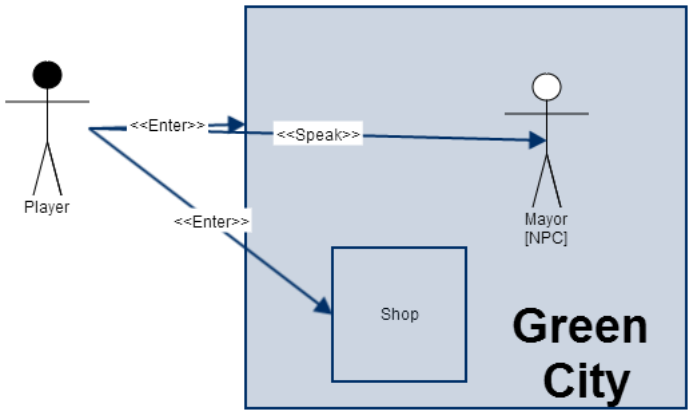


Figure 3.4: An example of a Diage diagram without using predefined actions

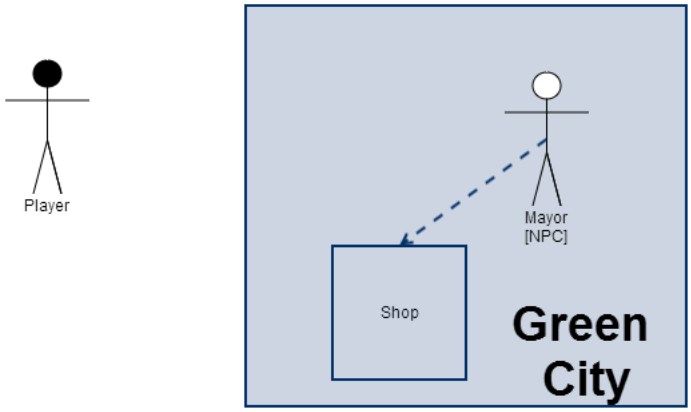


Figure 3.5: A *Diage* diagram showing a explicit connection between the Mayor and the Shop

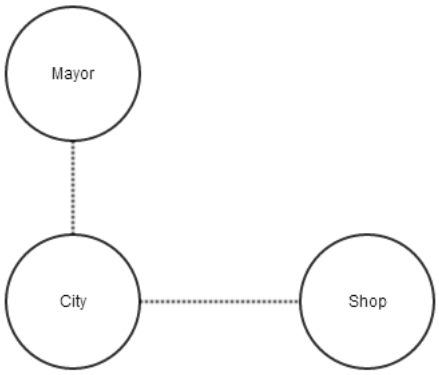


Figure 3.6: Figure 3.3 as a graph

Chapter 4

Narrative planning

This chapter will discuss the use of a narrative planner as used by many of my sources. I want to incorporate a narrative planner that smartly uses Diage to manipulate a narrative and conveys that in a way we petty humans can understand

A lot of related work as gone into the use of a planning system that decides on the narrative structure. Researchers like Riedl [6][5][7] and Cavazza [1] have been researching the use of artificial intelligence for years, and have made some interesting planning systems like *Memesis* [*citation needed*]. I am developing my own planning system to use *Diage* as input and output and use *Ludoscope* to manipulate the diagram as a graph (see section 5 for more on Ludoscope) for the model transformations.

Chapter 5

Usage in Ludoscope

This chapter will discuss the usage of Ludoscope for the procedural generation of plots and the diage knowledge model for a hypothetical quest generator.

Chapter 6

Further study

Bibliography

[1] Marc Cavazza, Fred Charles, and Steven J. Mead. Character-based interactive storytelling. IEEE Intelligent Systems, 17(4):17–24, July 2002.

[2] Marvin Katilius-Boydstun. The semiotics of A.J. Greimas: An introduction. Lituanian quarterly journal of arts and sciences, 36(3), 1990.

[3] Brian Magerko, John E. Laird, Mazin Assanie, Alex Kerfoot, and Devvan Stokes. Ai characters and directors for interactive computer games. In Proceedings of the 16th conference on Innovative applications of artifical intelligence, IAAI'04, pages 877–883. AAAI Press, 2004.

[4] Julie Porteous and Marc Cavazza. Controlling narrative generation with planning trajectories: The role of constraints. In Proceedings of the 2nd Joint International Conference on Interactive Digital Storytelling: Interactive Storytelling, ICIDS '09, pages 234–245, Berlin, Heidelberg, 2009. Springer-Verlag.

[5] Mark Owen Riedl, C. J. Saretto, and R. Michael Young. Managing interaction between users and agents in a multi-agent storytelling environment. In Proceedings of the second international joint conference on Autonomous agents and multiagent systems, AAMAS '03, pages 741–748, New York, NY, USA, 2003. ACM.

[6] Mark Owen Riedl and R. Michael Young. Character-focused narrative generation for execution in virtual worlds. In Proceedings of the International Conference on Virtual Storytelling, pages 47–56, 2003.

[7] Mark Owen Riedl and R. Michael Young. An intent-driven planner for multi-agent story generation. In Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1, AAMAS '04, pages 186–193, Washington, DC, USA, 2004. IEEE Computer Society.

[8] Nikitas M. Sgouros. Dynamic generation, management and resolution of interactive plots. Artificial Intelligence, 107(1):29 – 62, 1999.

[9] Peter William Weyhrauch. Guiding interactive drama. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 1997. AAI9802566.