Architecture document

Robert Kraaijeveld, Cees-Jan Nolen and Steven Schenk

8-09-2016

Contents

1 Introduction

In this document, we will give an abstract outline of the structure and architecture of our internship project. This will help us provide a more stable vision for the future of this project and will allow us to divide tasks between ourselves more easily. Since we cannot yet predict the exact form that this project will take, the architecture will be presented at quite a high level. We will also provide an an abstract view of the algorithms that we intend to use for the procedural generation of avatars with distinct personalities.

2 ERD Diagram

This very basic ERD Diagram describes the way in which we will separate the Unity-specific code, like providing proxy-properties to Casanova from the abstract Casanova code. In this way, the Casanova code for this project will be completely re-usable within a different framework like XNA or Unreal; only the proxies and the XML writer/reader code will have to be altered or rewritten.

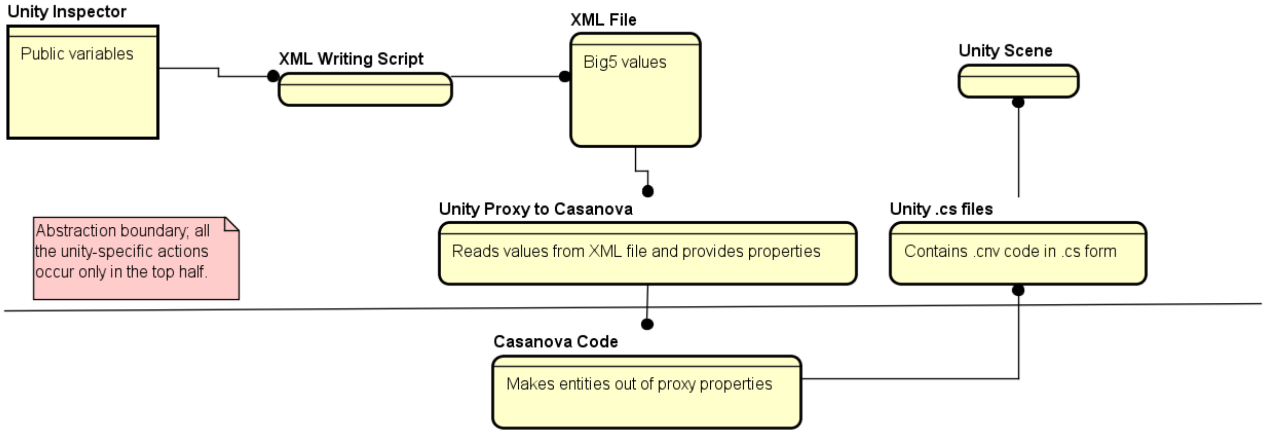
Unity Inspector
Public variables
XML Writing Script
XML File
Big5 values
Unity Scene
Unity Proxy to Casanova
Reads values from XML file and provides properties
Unity .cs files
Contains .cnv code in .cs form
Abstraction boundary; all the unity-specific actions occur only in the top half.
Casanova Code
Makes entities out of proxy properties

Figure 1: ER-Diagram

We will most likely provide the big-5 values (or other personality values, more on this later) through an XML file which can be manipulated through the Unity directly inspector. By using an XML file we will be easily able to bring certain personality configurations along on demos and manipulate these configurations from outside of Unity. An important point for us to consider is that the Unity proxies and all other Unity-related pieces of code should not deal with the actual logic of the game; They should only provide an easy interface for Casanova to interact with. By doing this our project will remain relatively independent of Unity, making it easier to use for possible future developers.

3 Algorithm structure for a single character

In the stages of this project, we will be focusing our attention on having a certain character behave according to his/her personality values and having this character interact with the player accordingly. Characters interacting with other characters is a much more complex subject; we will discuss this in a later chapter.

We have identified the most important visible behaviour-characteristics that each avatar should exhibit (almost) regardless of the context of the game in question. There are some exceptions to this rule however, depending on the nature of the game in question. In a relatively static game in which neither player nor npc move around much, like an interrogation-game, movement patterns and walking styles will naturally not be visible. The following list will include examples of each kind of behavior in order to make their distinctions more clear.

1. Decision-making (Will the character tend to the other NPC who has just fallen over or will I move away?)
2. Movement pattern and walking style (Does the NPC move at a steady pace or does he constantly change his walking speed? Does he walk about confidently or does he slowly wander?)
3. Body Language (What kind of mood does the NPC's body language reflect? Is he standing up proudly or are his shoulders collapsed?)
4. Speech (What does the NPC say and with what tone of voice does he say it? What mood does his voice reflect?)
5. Facial Expression (Which emotions does the NPC's face reflect? Does his facial expression stay neutral and constant or does it change often and rapidly?)

Each of these factors can provide a reasonably visible indication to the player of what kind of personality/emotions the character exhibits. To determine how a certain NPC will display these factors (IE. What the NPC's walking style will be) we will use a personality model. Personality models provide us with a measure of what kind of behavior an NPC who leans to certain sides of a psychological trait would exhibit.

3.1 Change in the behavior of an NPC

The behavioral factors of an NPC are changeable and can potentially change for two reasons: Because of the personality of the character itself, or because of certain events that happen to or near the character combined with that characters' personality. Certain personality traits contained in the earlier mentioned personality models, like neuroticism, can cause people to have very variable emotions and extreme emotional reactions to very insignificant events. Within the game, we will likely represent this by having characters with a high neuroticism level change their movement style, body language and facial expression constantly.

The other way in which we will let our characters behavioral factors change is by subjecting them to certain positive or negative events. Depending on the characters personality, they will respond differently to certain events. An example; NPC 1, who has an egoism value of 100 out of 100 stands next to NPC 2. All of a sudden, NPC 2 lets out a cry of pain and collapses. Because NPC 1 has such a high egoism value, he does not rush to NPC 2's aid: He completely ignores him instead and calmly walks away (An even more cynical reaction might be that NPC 1 starts searching NPC 2 for valuables..).

If NPC 1 had had a different egoism value, he/she would have reacted differently to this distressing situation. Positive situations will also elicit a certain response depending on the personality values of an NPC. When dance music suddenly starts playing next to a group of NPCs, the extravert ones among them might start to dance, whilst the NPC with very low extraversion scores probably won't dance.

Some of the behavioral factors of the NPC's will have to be slightly exagarrated in order for them to be clearly visible to the player and in order for them to be technically viable. Facial expressions in real human beings can be incredible subtle and complex, but emulating exact facial expressions would not only be an incredible technical challenge; it might take extreme player effort in order to actually notice them. Since our focus is still on creating a game, rather than a person simulator, we have prioritized 'playability' over realism.

3.2 Our personality models of choice

For our first prototype of a single avatar, we will be using the Big 5 personality model. This model has several layers of complexity, but for our very first prototype we will be focusing solely on the top level of the Big 5 model. This personality model containts, as the name implies, 5 principle traits by which to measure personality. They are as follows:

- Extraversion vs. Introversion
- Egoism vs. Altruism
- Thoroughness vs. Chaoticness [1]
- Emotional Stability vs. Neuroticism
- Openness to new experiences vs . Stubborness (Not supported) [2]

Each of the left-handed personality factors will be given percentage, ranging from 0 to 100 percent. A 100 percent score will mean that the NPC will totally exhibit the given personality factor of the Big 5 model, whilst a 0 percent score will indicate that the NPC will totally exhibit the opposite factor (In the case of Extraversion, this would mean that the NPC is totally introvert). In order to allow the user to create populations of NPC's without having to set each of their values individually, each personality factor will have both a minimum and a maximum percentage. These percentages will indicate between which percentages the scores for this personality factors will range for the NPC population.

In the future, we will likely replace the Big-5 model by more complex and larger personality models like HEXACO, in order to create more nuanced and refined NPC personalities. This would also require creating more distinct behaviors in order to accomodate the larger amount of personality factors contained within, for instance, HEXACO.

[1] In the original model, the term 'conscientiousness' is used. Since this term did not seem very descriptive to us, we replaced it with 'Thoroughness'.

[2] Item no. 5 is a little harder to implement as very concrete NPC behavior due to it's relevance lying mostly within the psychological growth and gathering of experiences, therefore we will not be using much it in our prototypes.

3.3 Algorithm structure

The algorithm which we will describe here will be responsible for deciding for each character what kind of behavior he/she will display in a given situation. This behavior will consist of a combination of facial expression, body language, speech [3] movement style and movement pattern and certain decision-making processes.

Before the actual algorithm can make decisions on an NPC's behavior, the following will need to be defined:

1. Define each NPC's personality scores, by choosing a random integer between the minimum and maximum value for each of the four personality factors.
2. For each of the personality factors, define behavior consisting combinations of animations, movement patterns, facial expressions, body language and speech. Define multiple behaviors for the extremes of each personality factor (IE. 0 percent extraversion and 100 percent extraversion) and preferably also some in-between behaviors. Define at what end of the spectrum each behavior lies. Note that you might have to define different behaviors for the same personality factors: Behavior for use in events and behavior for use when there are no events happening near the NPC.
3. Define a series of events that will be triggered throughout the duration of the game. For each of these events, take all of the four personality factors and give them a threshold value ranging from 0 to 100. The higher this threshold is for a given personality factor, the less likely that an NPC's reaction to this event will be driven by that personality factor.

Each frame, there will be a check to see wether the NPC is in the vicinity of an event. If so, the algorithm will do the following:

1. Get the greatest personality factor of the NPC that is greater than its' equivalent threshold in the event. If there is not a single personality factor of the NPC larger than its' equivalent threshold, stop: The NPC doesn't react to the event.
2. Get the event behaviors belonging to the resulting personality factor of step 1, if there is any result.
3. Determine which of the behaviors to trigger using the personality factors' value. (IE. a 100 percent egoism value would warrant a very egoistic behavior, etc.)
4. Trigger the behavior.

[3] Since we will most likely not have professional voice actors at our disposal, speech will probably consist of text bubbles or even just simple emotes like exclamation marks etc.

If on the other hand, there is no event within the vicinity of the NPC, every 5 seconds or so the algorithm will do the following:

1. Sum all of the personality factor values.
2. For each of the personality factors, compute the chance of the behavior relevant to this factor will be triggered: Divide the personality factor value by the result of step 1.
3. Order the personality factors by their chance values, starting with the biggest chance. (For instance: Egoism has chance value 30, so gets the first place, Extraversion has chance value 20, so gets the second place, etc.)
4. Generate a random number from 0 to 100: Get the personality factor in whose spot the number is. (For instance: If the resulting number is 40, extraversion will be chosen since 0-30 was covered by egoism, and 30-50 was covered by extraversion.)
5. Get the personality factor score, NOT the chance value, for the chosen factor and trigger the behavior relevant to that score.

4 Interaction between NPC's

In order to create a realistic feeling for the player, there also needs to be interaction between the different NPC's, rather than just single NPC's displaying behaviors. One way to do this would be in the form of mini-events between NPC's that only occur when NPC's get within a certain range of eachother.

A very basic event that might happen when two NPC's enter each others' personal space is that one of them might try to begin a conversation, if his personality scores [4] allow him to. Depending on the personality scores of the other NPC, he/she might accept the invitation. During the conversation, the body language, speech patterns and facial expressions of both NPC's will be governed by their relevant personality scores, like we talked about in the previous section.

Another simple interaction which would only involve 2 NPC's at a time would be navigation: When an NPC navigates a crowd of other NPC's, they might back away from him or hold their ground, depending once again upon certain of their personality scores. This very simple measure would create a more realistic representation of a crowd or group of actual humans.

We thought about including some slightly more socially unacceptable behavior into the mix, like fights and arguments breaking out between NPC's. However, we have decided to omit this kind of behavior for now because we think it might make it harder for the player to notice NPC personalities.

Group behavior is a subject which we are still looking in to: It might really enhance the realism of the game, but it will provide a great technical challenge and it might obscure individual personalities and behaviors in the same way that the aforementioned socially unacceptable behavior might.

[4] Within the Big-5 model, Extraversion vs. Introversion and Emotional Stability vs. Neuroticism are the most likely candidates for this task.

5 Demonstrating the NPC's

We intend to demonstrate this project by creating a game, or several games, in which player NPC interaction and NPC personalities will be key concepts. The simplest and probably most usefull kind of demo would be to have the player walk around a group of NPC's and have the player identify the NPC who exhibits a certain personality trait the strongest. This would be quite easy to implement: The only addition that is needed is for us to randomly select a certain trait, then store which NPC has the highest score on that trait. Finally, some controls for the player need to be added and the demo is ready. Because we use an XML file to store the ranges a certain personality trait may exhibit for a given NPC population, we can easily create multiple XML files storing various personality ranges and feed them as input to the game whenever we choose, providing us with an easy way of showcasing different NPC populations.

A more complicated demonstration game would be to have the player interact one-on-one with an NPC within the context of an interrogation game. A possible way to implement this would be to have a certain NPC being suspected to have committed a certain crime, with the player tasked to ask questions about this crime and decide, based on the NPC's responses, body language and facial expression wether the NPC has committed the crime or not. This would require some extra additions to the vanilla concept, like generating random crimes, their circumstances and the NPC's degree of involvement in the crime. It might also require some more advanced facial animations in order to create a believable 'suspect'. If implemented well enough however, it might provide a very entertaining experience to the players. It might also provide data on wether the NPC's personalities can also shine through and be believable in a one-on-one situation.

Finally, we might expand on the first kind of demo by including more of a challenge factor. This could be done by giving the player a time limit to find an NPC with a certain trait-extreme or by giving a 'profiler' role to the player. This would involve the player having to pick out an NPC from a crowd who is most likely to be agressive or commit a crime, by carefully examining their behavior. This task can be made more challenging by giving the player only one chance: If they choose the wrong NPC, they lose instantly. This might require a more detailed personality model than our current one, which only measures 5 of the primary personality traits of humans. HEXACO, for example, includes factors like fear of physical pain, anxiety, patience etc. which could all be potential characteristics exhibited by criminals who are about to commit a crime.

Another variation on this 'spot the criminal' idea could be having the character protect a VIP NPC; The player then has to pick out the non-VIP NPC most likely to harm the VIP, based on their behavior.

All of these examples can provide an entertaining game with the ability to tell us how well the characters emulate real personalities; the amount of time and resources we will have available when we have established a good baseline for procedural avatar personality generation will influence which demoes we will and will not implement.

We want to create each of these demoes as a game for the experience room, in order to provide a more lifelike interaction between the player and the NPC's, and in order to provide data on how well the concept of characters with distinct personalities works within the context of a mixed reality environment.