Utrecht University

Faculty of Science

Artificial Intelligence Master

Agents with Personalities

Developing agents with personalities used as Unity game characters

Author:
Ioana Cocu
s.n.4121651

Supervisor:
Mehdi Dastani

Second Examiner:
John-Jules Meyer

October 19, 2015

Universiteit Utrecht

Contents

List of Figures

List of Tables

Abstract

The creation of game characters with realistic behavior is an ongoing concern in the gaming and serious game industries. One way to achieve this is to find a way in which agent like characters can display human-like personality traits. The MBTI (Myers-Briggs Type Indicator) classification offers a way to classify human personalities using 16 types (e.g. INTP - Intraverted, iNtuitive, Thinking, Perceiving; ESFP - Extroverted, Sensing, Feeling, Perceiving; etc.) and 8 cognitive modes (NI - Intraverted iNtuition, SE - Extraverted Sensing, etc.). Each of the types consists of a combination of the 8 eight modes in a specific order. Although in psychology the personality types are considered descriptive and not prescriptive when they are studied in relation to job recommendations, for example, when this theory is adapted to autonomous agents, there is no distinction between these two approaches. In psychology, the specific order of the cognitive modes influences the behavior. In order to create a software model of this psychological theory, the importance of the cognitive modes in determining behaviors needs to be quantified. A quantified model is proposed and tested against pre-existing data (consisting of a list of job recommendations made for each of the 16 types). After it was determined that the proposed representation is valid, it was used to model the behavior of game characters in a game programmed in Unity using C#. By developing the game, I have shown that the BDI paradigm, specific to agent- oriented programming can be ported to an object- oriented language as C# and combined with psychological elements, in order to create game characters with more realistic behavior.

Chapter 1

Introduction

1.1 Background

The behavior of agents in multi or single agent systems can be described using the BDI(beliefs, desires, and intentions) logic, as presented by Bratman [27] and Cohen and Levesque [8] and Rao&Geoff [29]. This type of approach manages to describe and create behaviors for autonomous rational agents when implemented in agent oriented programming languages such as 2APL and 3APL [10] [11]. The drawback with these programming environments, however, is the lack of portability, compared to more broadly used object oriented programming languages, such as C# or Java.

Considering the interest that exists for the creation of game characters that exhibit realistic and diverse behaviors, the usage of design patterns that port the BDI approach to object oriented programming is a necessary step [12]. Moreover, in order to create human-like diverse behaviors in the characters, it is necessary to adapt cognitive psychology concepts to computer generated agents. For this purpose, the MBTI(Myers-Briggs Type Indicator) and Jungian [22] personality theory can be chosen because it offers a easy quantification of psychological traits in the form of eight cognitive modes that can be combined to create 16 personality types.

Design patterns that translate agent oriented programming strategies object oriented languages(in this case C# has been used) could be used together with an adaptation of Jungian psychological theory for obtaining realistic game characters in terms of the actions that they choose to perform. In this approach, personality theory becomes prescriptive (it is used to chose and predict future actions), although, when humans are concerned, it is considered as a descriptive trait (it only shows tendencies or explains past behavior and cannot predict anything). This drawback can be overcome by proving that, in a computer scientific approach, the two terms can become equivalent (what explains past behavior can also be used to predict future behavior).

In this thesis I aim at showing that it is possible to achieve realistic behavior in game

characters by using the MBTI psychological types as models for the characters and by adapting the BDI paradigm to an object oriented programming language while using the Unity programming platform as a tool for game development.

1.2 Problem

In this project, it will be shown that it is possible to adapt the MBTI personality theory, alongside with constructs that are specific to agent oriented programming (such as BDI logic) to object oriented programming languages such as C#. The proof for this consists of a game that shows multiple autonomous characters that exhibit different behaviors based on the personality type they are assigned by the player at the beginning of the game.

The research questions that will be answered within this thesis consist of:

- How can computer generated characters show realistic behaviour?
- What are the main differences between the agent oriented programming languages and object oriented ones in terms of expressiveness and how can they be overcome?
- What makes a game entertaining for the player?
- Is there a way to represent the psychological characteristics in a simple flexible manner? If so, how can it be demonstrated that it truly corresponds to the original description?
- What is the best programming solution for the implementation of agents, using object oriented programming(C#) and also integrating the characteristics of the 16 personalities, after they have been quantified in a way that is more usable by computer science?
- What are some conclusions that can be drawn from the whole implementation and how could it be improved in the future?

1.3 Methodology

In order to answer these questions, a literature study will be done, that covers: the psychological approach towards personality theory and the MBTI classification, some general information about computer games and game design, agent oriented programming versus object oriented programming and design patterns, and an overview of the Unity programming platform. After the literature review a manner of implementing the 16 personalities as defined by the MBTI will be proposed. This implementation will be tested using pre-existing data, consisting of a list of job recommendations that were made for each of the types. The initial list will be reconstructed, so that cognitive modes are the ones that

determine the degree with which a job recommendation is suitable for each personality type. The results will be compared to the initial data, in order to test if this approach of implementation is viable.

After that, the personality implementation will be integrated in a solution that determines the behavior of autonomous non user controlled game characters, using the C# programming language and the Unity platform for implementing game specific actions (e.g. game physics). The game that showcases this implementation will be described within the thesis and then conclusions will be drawn about the differences in behavior between characters with different personalities and how they can observed visually during the game play.

1.4 Outline

The thesis is structured as follows: the first chapter consists of a short introduction that states the research questions that the thesis aims at answering, followed by a chapter containing a literature review that concentrates on personality, agents and design patterns materials and a short description of the Unity programming platform, the next chapters describe implementations of the theoretical ideas presented in the literature - firstly an experiment that proves that the MBTI classification can be correctly implemented using only the succession of cognitive modes in a way that is specific to each personality type, and secondly, a description of the game that was created as a final result of the project and how the autonomous characters were created, the last chapter of the thesis consists of a conclusion that synthesizes all that has been accomplished and possible future work that can be done to further the research.

Chapter 2

Literature Review

2.1 Personality theory and MBTI

The 16 MBTI personalities, as first described by Katherine Briggs and Isabel Myers consist of four items: two "functions" and two "attitudes" that people exhibit when they make decisions. According to them, a hierarchy of the ways in which cognitive modes manifest within each person can be observed and used to understand the way in which humans act.

In the next sections, the characteristics of the psychological types and cognitive modes and the way in which they influence behavior will be explained, as well as the way in which a hierarchical order of the cognitive modes can be constructed for each type and how they correspond to the Jungian archetypes.

2.1.1 Introduction to Personality Theory

Carl Gustav Jung distinguished between what can be considered "functions" and "attitudes" of consciousness [22]. In his book *Personality Theory* he only explicitly stated the Extraversion/Intraversion attitude pair. According to him, extraversion is the flow of energy towards the exterior world, while intraversion draws the psychic energy towards one's own interior. The two function pairs that were defined in the same book are:

- the "perception" function with 2 instances - Sensing(perception is obtained through the sense organs) and Intuition(perception is obtained through the unconscious)
- the "judgement" function with 2 instances - Thinking(intellectual cognition and the forming of logical conclusions) and Feeling(subjective valuation)

Based on these two function pairs and the attitude pair, the cognitive modes are determined as a combination of a function instance and an attitude instance (i.e. Extraverted Feeling - FE, Intraverted Feeling - FI, Extraverted Thinking - TE, Intraverted Thinking - TI, Extraverted iNtuition - NE, Intraverted iNtuition - NI, Extraverted Sensing - SE,

Intraverted Sensing - SI). Each person can be described through an order and frequency of manifestation of eight cognitive modes that are composed of a couple attitude-function. This order and importance of the cognitive modes can be used to explain preferred behaviors and attitudes of humans. The first two cognitive modes corresponding to each personality type, named the dominant and the auxiliary cognitive mode play the most important role when defining a person's consciousness and behavior.

In Jung's version of personality theory, a personality could be expressed using 3 components - attitude instance (Extraversion/Intraversion), perception function instance (Sensing/Intuition), and judgement function instance(Thinking/Feeling). Jung stated that the dominant and auxiliary cognitive modes are determined by the perception and judgement functions' instances that correspond to the subject in combination with contravert attitudes (i.e. if the dominant cognitive mode has Extraversion as an attitude instance, the auxiliary will have Intraversion as an attitude instance, and the other way around). The dominant attitude instance is the one that primarily defines the person's character. The problem that Jung encountered was that the function instance that corresponds to the dominant function could not be clearly established using only one pair of attitude instances.

To illustrate this problem, an example of a someone whose personality is defined as ESF (Extraverted, Sensing, Feeling) will be examined. As previously stated, the dominant cognitive mode will have Extravert as an attitude instance and the auxiliary cognitive mode will have Intravert as an attitude instance. These attitudes need to be paired with a function instance, in this case Sensing (as an instance of the perception function) and Feeling (as an instance of the judgement function). Jung's theory cannot establish which of the two cognitive functions will be used for the dominant and which for the auxiliary cognitive modes.

In order to be able to accurately define the way in which the dominant and auxiliary modes are defined, Briggs and Myers introduced a new attitude pair: the "lifestyle" attitude that consists of the Perception and Judging instances. This pair is usually represented as P-J so it is not to be confused with the names of the Jungian functions. The additional pair is used so the dominant and auxiliary cognitive modes can be determined when the personality of a subject is described using both attitude pairs, alongside with the function pairs.

The dominant cognitive mode can be obtained as follows, based on a 4 letter MBTI personality type:

- If the first attitude instance is Extraverted
  - If the lifestyle attitude instance is P - the dominant cognitive mode will be E+the instance of the perception function (Sensing/iNtuition)
  - If the lifestyle attitude instance is J - the dominant cognitive mode will be E+the instance of the judgement function (Thinking/Feeling)
- If the first attitude instance is Intraverted

- If the lifestyle attitude instance is P - the dominant cognitive mode will be E+the instance of the judgement function (Thinking/Feeling)
- If the lifestyle attitude instance is J - the dominant cognitive mode will be I+the instance of the perception function (Sensing/iNtuition)

For the auxiliary cognitive mode, the contravert of the dominant cognitive mode's attitude is used alogside with he instance of the function that was not used in the dominant cognitive mode (i.e. if the dominant cognitive mode uses the perception function instance, the auxiliary will use the judgement function instance).

A personality group consists of 4 MBTI personality types that are considered similar in terms of behavoir. Each of the 4 groups has 2 pairs of cognitive modes that represent the dominant and auxiliary cognitive modes for any of the MBTi personality types that belong to the group.

A list of all the MBTI personality types, the group they belong to and their dominant and auxiliary cognitive modes can be seen in Table 2.1

Table 2.1: Types, Modes and Personality group [38]

Group	**MBTI Type**	**Cognitive Modes**	**Group**	**MBTI Type**	**Cognitive Modes**
Sentinels	ISTJ	SI, TE	Diplomats	INFJ	NI, FE
	ISFJ	SI, FE		INFP	FI, NE
	ESTJ	TE, SI		ENFJ	FE, NI
	ESFJ	FE, SI		ENFP	NE, FI
Explorers	ISTP	TI, SE	Analysts	INTJ	NI, TE
	ISFP	FI, SE		INTP	TI, NE
	ESTP	SE, TI		ENTJ	TE, NI
	ESFP	SE, FI		ENTP	NE, TI

The personalities that belong to a certain group are considered similar from a behavioral point of view and exhibit similar characteristics [6]. It can be observed from the table that within a group there are two pairs that have the same dominant and auxiliary functions, but in reverse order.

2.1.2 Cognitive modes

The MBTI Personality types are useful when subjects are tested or when it comes to describing behavior and dividing the population into categorizes, but the characteristics that they represent are actually the cognitive modes and the way in which they manifest within each person. The MBTI can be considered a good tool for deciding in which order

and how the characteristics of the cognitive modes are bound to manifest within each person's type of consciousness.

The Cognitive modes and their characteristics can be seen in the following table (Table 2.2):

Table 2.2: Cognitive modes description [37]

Cognitive mode	Essential value	Description
SE	Experiment	Discovers new ideas and phenomena by direct experience
NE	Ideation	Rearranges known concepts into novel systems
TE	Organization	Efficiently manages resources, decisive, imposes structure
FE	Community	Expressive, tactful builder of group morale
SI	Knowledge	Physically self-aware, values practice and technique
NI	Imagination	Prophetic, guided by inner fantasies and visions
TI	Analysis	Logically improves rational performance
FI	Evaluation	Uses personal values to distinguish good from bad

In Table 2.2 each of the eight cognitive modes is presented in terms of its "Essential value" - a noun that describes what a person that mannifests that cognitive mode yearns for in general terms, and "Description" - where some behaviours that are associated with that cognitive mode are mentioned.

2.1.3 Types and archetypes

Although in the previous subsections only the dominant and auxiliary cognitive modes have been discussed, all the 8 cognitive modes are present in each of the MBTI personality types, but their role and influence upon the behavior of the individual is dependent on the MBTI type to which he belongs (e.g. The cognitive mode succession will be different for a ENTJ - Extraverted, iNtuitive, Thinking, Judging, and a ISTP - Intraverted, Sensing, Thinking, Perceiving).

The manner in which the dominant and auxiliary modes are determined for each of the MBTI types has been discussed in the previous sections of this chapter. For the other 6 modes, Table 2.3 can be used as a reference in how to obtain them for each of the 16 personality types. A complete list of the order and importance of cognitive modes for each of the 16 MBTI personality types and their correspondence to the Jungian archetypes can be seen in Appendix A of the thesis.

Table 2.3: Archetypes and cognitive modes

Archetype	**Cognitive mode**
Hero	Dominant
Parent	Auxiliary
Child	Auxiliary Opposite
Anima	Dominant Opposite
Opposer	Dominant Contravert
Witch	Auxiliary Contravert
Trickster	Witch/Senex opposite
Daemon	Opposer Opposite

Table 2.3 shows how the complete order of cognitive modes can be obtained for each of the MBTI types. To clarify what the cognitive modes description from the table signifies, it is easier to use an example. For someone who belongs to the ESTP MBTI type, the cognitive modes importance is as follows:

- Hero - Dominant- SE (it has been explained in the previous section how the dominant cognitive mode is obtained)
- Parent - Auxiliary - TI (it has been explained in the previous section how the dominant cognitive mode is obtained)
- Child- Auxiliary Opposite - FE (the auxiliary in this case was TI, so its opposite will be the judgement function instance opposite - Feeling in this case + the contravert of the auxiliary's attitude - Extravert)
- Anima - Dominant Opposite - NI (the dominant in this case was SE, so its opposite will be the perception function instance opposite -iNtuition in this case + the contravert of the dominant's attitude - Intravert)
- Opposer - Dominant Contravert - SI (the dominant in this case was SE, so the perception function instance will remain the same -Sensing in this case + the contravert of the dominant's attitude - Intravert)
- Witch - Auxiliary Contravert - TE (the auxiliary in this case was TI, so the judgement function will remain the same - Thinking in this case + the contravert of the auxiliary's attitude - Extravert)
- Trickster - Witch/Senex opposite - FI (the witch in this case was TE, so its opposite will be the judgement function instance opposite - Feeling in this case + the contravert of the witch's attitude - Intravert)

- Daemon - Opposer Opposite - NE (the opposer in this case was SI, so its opposite will be the perception function instance opposite -iNtuition+ the contravert of the opposer's attitude - Extravert)

The names and characteristics of the archetypes connected with the cognitive modes was first suggested by Jung's theory that proposed that archetypes were influencers of human behavior in a similar manner in which the ego played a role in Freud's theories. Jung's archetypes were adapted to the MBTI [2].

The main influence of each of the archetypes is shown in Table 2.4, with the first four archetypes being the ones that influence the behavior the most.

Table 2.4: Description of the influence of the archetypes

Archetype	**Description**
Hero	Masterful
Parent	Helpful
Child	Playful
Anima	Spiritual
Opposer/Dreamer	Undermining / Speculative
Witch / Conserver	Obstructive / Resolute
Trickster/Jester	Deceptive /Amusing
Daemon/Daredevil	Dangerous /Rash

The last four archetypes are called the "shadow archetypes" and they can be perceived as a positive or a negative influence in terms of impact upon the behavior of a person. The characteristics of positive and negative shadows [38] can be seen in the second half of Table 2.4, alongside with their corresponding names.

An archetype influence changes from negative to positive based on the personal development, life experiences and age of a subject [38].

Within this thesis, the effect of the archetypes (as extra variables determining behavior) will not be taken into consideration and all shadows will be considered positive. They will be used though, when describing the importance and order in which the cognitive modes manifest for a certain MBTI type.

2.1.4 MBTI Types and career choices

The distribution of the MBTI personalities and stability of MBTI type that corresponds to an individual has been researched since the classification was introduced and it has been shown that, once a person has been assigned to a personality type through testing (usually

the tests consist of a list of sentences with which the subject agrees or disagrees, with the objective of observing what instance of the 2 functions and 2 attitudes best describes him or her), the results are stable when all four indicators (E/I, S/N, T/F, P/J) are considered (one of the indicators might change, but, on the whole people obtain similar results even when tested six years after the initial classification) [7]. It can be argued that, although the classification by itself is stable, the fact that people might be reclassified when retaking the test might actually have to do with the way the test is administered and the fact that it does not take into consideration local norms that apply to different populations [3].

The relation between personality types and occupations has been studied. For example the frequency with which certain personality types are connected to people working in software engineering [5]. This study shows that there are more sensing (67%) than intuitive types and significantly more thinking (81%) than feeling types, and also slightly more introverts than extroverts and slightly more judging than perceiving types.

The influence of personality types has been also studied in the context of learning and the way in which it affects different student approaches towards learning [17], highlighting differences in their preferred types of studies (abstract vs. practical), working in groups vs. individually, self ratings in regard to their ability to solve problems creatively and the size of companies in which they would end up working in. Belonging to a personality type has also been shown to play a role when founders and CEOs of successful companies are concerned [18], with most of them belonging to the INTP (Intraverted, iNtuitive, Thinking, Perceiving), ISTJ (Intraverted, iNtuitive, Thinking, Judging), ENTJ (Extraverted, iNtuitive, Thinking, Judging), ENTP(Extraverted, iNtuitive, Thinking, Perceiving) and INTJ (Intraverted, iNtuitive, Thinking, Judging) categories for the fastest-growing companies in the United States in the late 1980s.

It has also been shown that the personalities of employees of a company are correlated to the personality type of their managers [33], a fact determined by the homogeneity of personality hypothesis that states that people would tend to stay in a workplace where their colleagues have personalty traits that are similar to their own.

The formation of teams that need to work together longterm at workplaces also benefits if MBTI testing is used when deciding upon the members that would be in their componency [4]

2.1.5 Other personality models

The big five

Other methods of personality classification include the five-factor model [14]. Introduced by William McDougall in 1932, this model divides the personality in five factors: intellect, character, temperament,disposition and temper [25] . The five factor model, in its modern form divides the personality into five characteristics: Extraversion/Intraversion, Friendliness/Hostility, Consciousness, Neuroticism/Emotional Stability and Intellect. These char-

acteristics are independent from each other and can be tested for separately. Just like in case of the MBTI, the scores tend to be stable in time for the majority of subjects.

The biggest drawback for this classification is the fact that the scores obtained for each of the traits only represent the way in which the subject responds to it, and does not imply any relation with the others - there is nothing as the succession of the same cognitive modes for the MBTI. In the case of the MBTI, although the scores obtained for each of the attitudes and functions are independent from each other, they are connected to the cognitive modes. This connection makes the MBTI classification more flexible and modular, so it matches better the scope of this project.

The enneagram

The enneagram model has been introduced by Oscar Ichazo and Claudio Navajo. This model presents 9 personality types that are described by their characteristic role, ego fixation, holy idea, basic fear, basic desire, temptation, vice, virtue, stress and security [21]. The personality types are commonly represented in a circle and each of them is linked to its immediate neighbors and other 2 types with which it shares some common characteristics. The personality types determined by the enneagram have been showed to be stable in time and a connection has been established between the types determined by the enneagram and the ones that are established by the MBTI [36].

The drawback of using the enneagram as the psychological basis for this project is that, similar to the big five classification the characteristics of the types are individual and there are few common features for all types. Another minus is that the classification is more complicated than the MBTI, without offering extra benefits and, what is more, there is limited literature available about the relation between the enneagram classification of a person and the manner in which they act in their lives, expressed in a quantifiable manner (such as career choices).

2.2 Agent Programming

This section explains the main characteristics of autonomous agents, the principles of BDI logic and how they are applied in agent oriented programming languages. The main differences between agent oriented programming and object oriented will also be reviewed, followed by a last sub-section explaining how the paradigm of agent oriented programming can be adapted to object oriented programming.

2.2.1 Autonomous Agents

Autonomous agents are used in a variety of environments, from gaming to industrial robots performing tasks that require autonomy. For each of these applications, the requirements

are very different on a superficial level, but there are some programming principles that need to remain constant. Some of these desired characteristics for agents are: [39]

- Autonomy - the agent is at least partially independent
- Local views - the agent does not necessarily have a global view of the whole environment, it only has access to local information
- Decentralization - there is no controlling agent in case of a multiagent system.

The environments in which these agents evolve are classified as follows [39]:

- Accessible vs. inaccessible - describing the ability of the agent to know all the necessary information about the surrounding environment.
- Deterministic vs. non-deterministic - describing the effect of actions taken; in a deterministic environment each possible action has a known outcome.
- Static vs. dynamic - describing the causes that change the environment, only the agent's action, vs. actions of other entities
- Discrete vs. continuous - describing the finitude of actions that can be taken; in a discrete environment, this number is fixed and finite.

In order to function correctly, and to be considered intelligent and autonomous, an agent has to have some capabilities [39]:

- Reactivity - the ability of an agent to sense the changes in the environment and respond to them accurately and at the right time
- Proactiveness - the ability to have a goal directed behavior, by taking initiative
- Social ability - the ability to interact with other agents or humans to satisfy certain objectives.

It is argued that autonomous agent programming is hard to attain through the object oriented programming paradigm (such as the one present in C#) [39]. Three main reasons are offered to justify this claim:

- autonomy - agents decide for themselves which is the best action they could take at a certain time, this degree of autonomy is harder to obtain with software objects;
- flexibility - the behavior of agents is more flexible than the one that is traditionally obtained through objects;
- concurrency - multiple threads are used for agent deliberation, while OOP traditionally has a single thread for execution.

These shortcomings can be overcome when software objects are customized to work similar to the way agents do.

The next sections will offer a more detailed view of agent oriented programming languages, the BDI logic and how they can be adapted to OOP.

2.2.2 BDI formalism

BDI(Belief Desire Intention) logic is the theoretical framework on which most agent-based programming languages are built [26] . According to Michael Bratman (Intention, Plans and Practical Reason) [27], intention plays a very important role in choosing a behavior in both human beings and agents. Intentions are based on the idea of pro-attitude - an agent's mental attitude directed towards an action under a certain description.

The intentions, according to Bratman [27], [26] have the following characteristics:

- They are high level plans
- The intentions are the ones that guide deliberation for creating more concise plans
- They are related to the notion of commitment - when an agent decides it has an intention
- They can be abandoned by an agent under one of the following conditions:
  - the intention has been achieved
  - the intention is impossible to achieve under its current beliefs
  - the agent has committed to another intention for which the previous one is instrumental

Cohen and Levesque [8] define the concept of intention as the combination of a persistent goal for which the agent has a belief that it is achievable. In its turn, a goal is persistent if it consists of an achievement goal (a goal that is currently not achieved, but which is preferred by the agent) that is kept until it is achieved or until it becomes in-achievable.

Rao and Geoff [29] provide another logic formalization of the BDI concept, using state formulas and operators such as: always, inevitable, optional, done, succeeded and failed.

2.2.3 Agent Oriented programming languages

Based on these logical formalizations of BDI, the programming languages 3APL(Artificial Autonomous Agents Programming Language) [20] and 2APL(A Practical Agent Programming Language) [10] [11] were developed.

The 2APL programming language specifies agents' characteristics in terms of:

- Beliefs - the beliefs implemented by the belief base

- Goals
- Basic Actions - there are 6 types of basic actions that can be performed by the agent:
  - belief update actions - that change the beliefs of the agent and are specified in terms of pre and post conditions
  - communication actions - that pass messages to another agent
  - external actions - that affect the environment of the agent
  - abstract actions - that encapsulate plans into a single action
  - belief and goal test actions - test whether an agent holds a certain belief
  - goal dynamics actions - consisting of adopting and dropping goals
- Plans - consisting of basic actions and process operators
- Reasoning Rules - used for the implementation and generation of plans
  - Planning Goal Rules - implements a plan if an agent has certain beliefs
  - Procedural Rules - generates plans as a response to messages from other agents or events in the environment
  - Plan Repair Rules - indicates what plan should be followed if a the plan that was currently executed by the agent should fail.

2.2.4 Representing agents with object oriented programming

The representations used for describing agents in programming languages such as the ones described above are useful for creating models of multiagent systems and understanding how the decisions process of an autonomous agent takes place. The main problem that is encountered when using these programming environments is the limited portability compared to the one of object oriented programming languages.

Object oriented programming languages, such as C# or Java can be run on a higher number of platforms and devices and, therefore, an adaptation of the agent oriented design to the object oriented paradigm is desirable [12]. The adaptation to Object Oriented design can be done if it respects the principles stated above concerning the necessary conditions for an agent's autonomy.

The beliefs of the agent can be set and changed using methods of type get() and set(), while the rules for implementing goals can be tested with logical constructs present in object oriented programming languages. Concurrency and deliberation on plans can also be achieved by using appropriate methods.

Given the wide flexibility of Object Oriented programming, it is feasible to say that most functionalities that are specific to agent programming can be achieved through the usage of the right programming design patterns.

2.3 Games and Gamification

In this section the concept of a game in a general sense will be explained. After that, different classifications of computer games [9], [15] will be reviewed, alongside with explanations about what makes a game design better in terms of gamification [13], [16], [30] and player experience. In the last part of the section, the usage of AI techniques in games in terms of game character creation will be examined [23], [24], [28].

2.3.1 Games and Computer Games

A game is defined as a closed formal system that subjectively represents a subset of reality [9], meaning that a game needs to have a set of explicit rules, is composed of parts that interact with each other and it does not need to represent anything from the objective world. The objective vs. subjective approach is what makes the difference between a game and a simulation or a training, in which the player needs to learn something that would have applications in the real world.

The three key elements of a game consist of: interaction, conflict and safety. Interaction is considered to be a key element that distinguishes a game from a story or from solving a puzzle [9] - the player needs to be able to obtain different outcomes based on his or her performance and to receive a form of feedback from other players or elements inside the game. Conflict means that players need to compete for a purpose (e.g. obtaining a higher number of points). Although cooperation might appear (e.g. players cooperate within their team and teams challenge each other), an element of conflict is crucial for games [9]. Safety implies that losing in a game is the lowest reward a player can get by playing, but is still a positive reward, therefore making games "a safe way of experiencing reality" [9].

Based on these principles 5 types of games can be distinguished:

- Board games - consist of a board surface and movable pieces that are controlled by the players, the players are concerned with the number and geometrical relationships between the pieces.
- Card games - the players use a deck of cards with different symbols and the rules state how different combinations of cards affect the outcome of the game, the players are concerned with the analysis of different possible combinations.
- Athletic games - emphasize the players' physical abilities and need interaction in order to be considered games, otherwise they are just competitions (e.g. a race is a competition and not an athletic game).
- Children's games - usually involve social skills and groups, containing simple mental and physical components.
- Computer games - involve interaction with a computer and usually emphasize hand eye coordination, although they can cover a wide range of genres and requirements.

There are many motivations why a person might chose to play a game, the most important ones being considered to be: exploration, the desire to prove oneself, exercise and the need for acknowledgement. All of these reasons play a role when a person decides what game/type of game they would play, alongside with the individual tastes of each person.

2.3.2 Game design and types

A general classification of computer games is proposed by Crawford [9] who divides them into 2 main categories, namely, skill and action games and strategy games. Each of these categories contains multiple sub types that will be enumerated and explained below.

Skill and action games are characterized by real time play and require hand-eye coordination as a primary skill of the player. This category can be divided into:

- Combat Games - there exists a direct, violent confrontation where the human player needs to shoot or use other means of defeating other players or game characters.
- Maze Games - a character controlled by the human player must move through a maze of paths while collecting objects or avoiding enemies (PacMan is a classic game that belongs to this category).
- Sports Games - they mimic the rules of real sports, but with game characters instead of human players (e.g. the Fifa games).
- Paddle Games - the player controls a paddle that collides with moving objects and changes their trajectory for a required goal (Pong is a classic example of a paddle game).
- Race Games - the player competes with another player or a computer controlled character towards a certain goal, usually reaching a place in the game space without breaking any of the established rules (e.g. touching objects that are assigned to the other player).
- Miscellaneous Games - are games that cannot be clearly assigned to any of the above categories, showing characteristics that belong in equal measure to more than one type.

Strategy games differ from skill and action games as they require more planning and less motor skills from the part of the player. Strategy games are divided into:

- Adventures - are games where the player needs to navigate through complex worlds, collect objects, overcome obstacles, etc. until they reach a determined goal.
- D&D Games - are also called fantasy or role-playing games and they are more concerned with the idea of a player assuming the identity of a certain character and exploring the game space than simple adventure games.

- Wargames - are based on old war board games, but have evolved in time in a similar manner in which computer sports games did from real life sports games.
- Games of Chance - are mostly based on card games
- Educational and Children's Games - although it can be argued that all games can have an educational component, this category explicitly aims at making the player learn a certain skill or information by playing.
- Interpersonal Games - concentrate more on the relationships between the game characters (e.g. what makes them cooperate with each other) than on the action.

The elements that compose computer games have to be studied independently from the ones that are specific to other entertainment mediums, such as movies, theater or story telling. Eskelinen [16] considers that there are six key elements to be considered when designing a computer game: order (the relation between system events and player events and how each of them can influence the game), frequency (how often certain events happen during the game play; the events can be automatic or require player involvement and action), speed (tha pace of the game), duration (the time limit for a game play), the time of action (the possibility of the player to act at certain moments during the gameplay) and simultaneity (the capacity to have multiple parallel events within a game play).

A more detailed classification and description criteria for computer games is considered by Elverdam [15]. He proposes that computer games could be classified based game space characteristics, the relationship between game playing and time, player composition and relation, struggle and game state. Each of these categories provide additional sub elements according to which games can be divided into types.

According to game space characteristics, the possible descriptors are:

- Perspective - describes if the player has an overall view of the gamespace (omnipresent), or only its close vicinity environment (vagrant);
- Positioning - describes whether the player is able to tell his exact location within the gamespace (absolute) or can only make guesses about it based on other objects that are visible (relative)
- Environment Dynamics - determines if the player is able to make alterations to the game space at any time (free), only in certain locations (fixed) or none at all (none).

External time consists of the following:

- Teleology - describes if the game ends at a given time (finite), or can go on forever(infinite);
- Representation - describes the passage of time within the game and if it reflects the way in which time passes in the real world (mimetic), or has its own rules (arbitrary).

Time can also be regarded as internal time is chosen as the independent variable, then the following descriptors appear:

- Haste - the passing of real time alters the game state (present) or not(absent)
- Synchronicity - agents act at the same time(present) or take turns(absent)
- Interval control - players decide when the next game cycle begins(present), or they cannot do it(absent)

The player composition describes the types and relationships between the players involved in the game - single or multi player, single or multi team, etc. The player relation can be viewed in terms of :

- Changeability - the relation between two players can change(dynamic) or not(static)
- Evaluation - the players can be evaluated individually, or as a team, or in both manners.

The struggle element of a game consists of:

- Challenge - describes how a game can provide opposition - predefined elements that remain the same every time the game is played (identical), variation of game elements through mathematical randomness (instance) and autonomous game agents (agent);
- Goals - that can be exact and unchanging for obtaining a victory (absolute) or can be subjective depending on the occurrences in a specific game (relative).

Finally, the game state is composed by:

- Mutability - describes how long a change in the game (performed by either the player or an autonomous game agent) state lasts - they can be passing(temporal), last throughout the game(finite), or continue through multiple game instances(infinite);
- Savability - describes if the player is able to save and restore any game state that he had reached (unlimited), this is only possible for certain predefined states(conditional), or it is impossible to save any game state(none).

2.3.3 Gamification

The term gamification has been introduced in the early 2000s to describe elements that are considered specific to games and the possibility of using them in other areas or activities, such as apps, simulations [13] and educational software [30]. This has helped to identify what are the parts that constitute a "gamefied" application:

- gamefulness - the experiential and behavioral quality (i.e. the experience of playing a game, even if the final goal is learning something, solving a real problem, etc)

- gameful interaction - artifacts affording that quality (interactions with other players or elements within the application)
- gameful design - usage of game design elements (e.g. objects, menus, sound, etc.)

Ten elements are considered to be the main components of successful game design [13]: self-representation with avatars, three-dimensional environments, narrative context, feedback, reputations, ranks and levels, marketplaces and economies, competition under explicit and enforced rules, teams, parallel communication systems that can be easily configured, and time pressure.

Considering these elements, it has been argued that gamification can be used as a tool in game design, in order to create better and more entertaining games [13].

2.3.4 Games and AI

AI can be used within games in order to create realistic behaviors in a variety of types of games (from chess to MMORPGs) [32]. The usage of AI in game development is of particular interest when autonomous game characters are concerned [23], especially when human like behaviors are concerned. It has been shown that humans tend to pay more attention and engage more when they interact with game characters that behave similarly to the way in which a human player might [24]. The way in which game characters behave has been studied mostly from the point of view of machine learning (especially adaptive learning through genetic algorithms) [28]. The adaptive learning algorithms are used to make the autonomous characters act based on what the actions that the human controlled character/s chose to perform. Machine learning is used to assure diverse behaviors from the non-playable characters within the game, that cannot be easily anticipated by the human player.

In this thesis the main focus will be the basic structure of an algorithm than can adapt the structure of the BDI approach towards autonomous agents to game play, and less on machine learning strategies, but this topic can be considered for future work.

2.4 Unity

The Unity package [34] [35] is a game-creation system that includes a game engine and an IDE (integrated development environment). This software is the choice of many game developers because it assures that the games created with it are portable to a range of devices - PC, mobile, etc. Another important feature is that the games created in this environment can be either 2D or 3D, with resources created specifically for each category.

The Unity environment is a very useful tool for programmers because it has pre-implemented libraries that create realistic movement and physics effects, so there is no necessity to concentrate on recreating and recalculating game physics elements.

An advantage of Unity as a game development environment is that the game design can be done separately from the programming and connected to it afterwards. The Mono development IDE, that corresponds to Unity offers the possibility to use scripts written in multiple programming languages - C # , JavaScript and Boo.

The graphical elements of the game can be created separately, with other software products such as Photoshop, Gimp, 3DSMax and then imported as sprites to compose the game elements or background. Any game element can constitute an object of type GameObject from the MonoBehaviour class, making the code easy to split into work modules and assuring that behavior of each element is independent.

Apart from the technical advantages that it offers, the fact that it is extensively used by many developpers assures that there is available documentation regarding the correct use of its features and common problems that might appear during the development stages of a project.

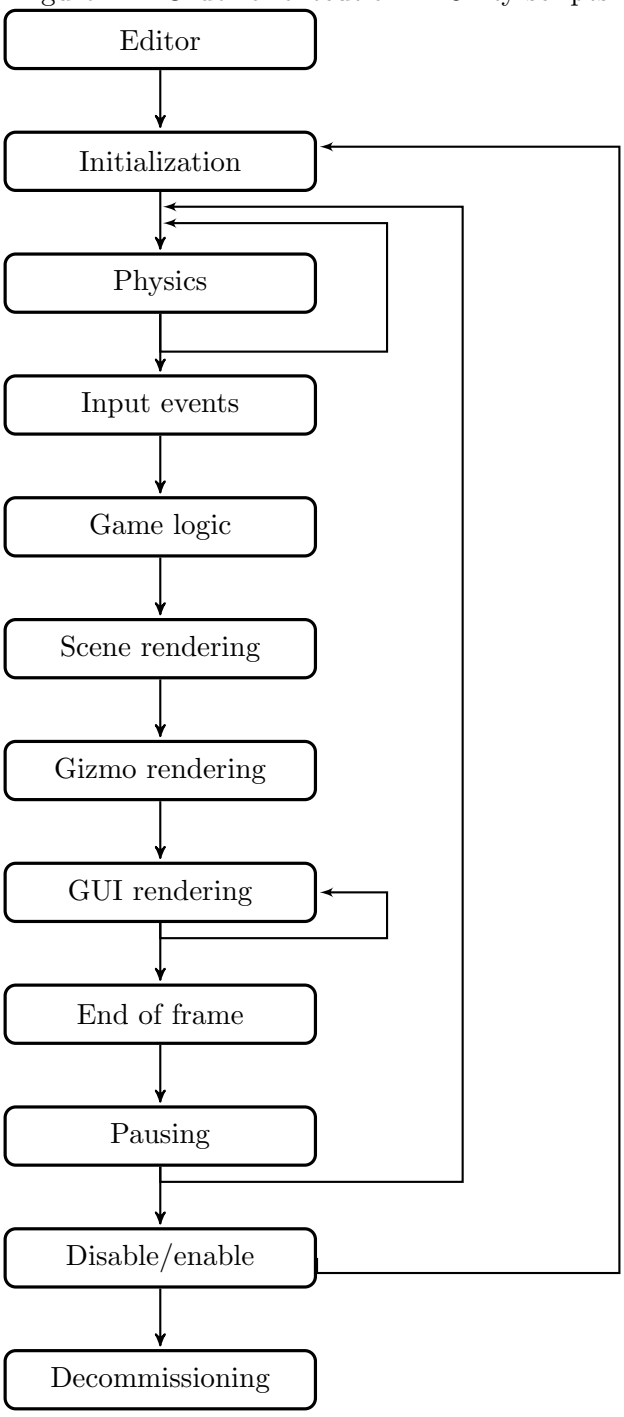
2.4.1 Script structure in Unity

The general structure of a Unity script can be observed in Figure 2.1. The elements of Figure 2.1 are then explained in detail in figures 2.2 through 2.13 (the routines that make up a module are shown in the order in which they are called on the right hand side of the module name).

It can be observed that the structure of a Unity game object is highly modular and the pre-implemented routines help dealing with many game design elements in a simpler manner.

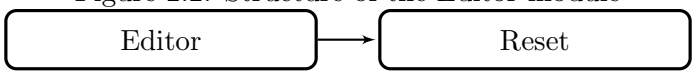
The Game Logic block of methods (Figure 2.6) are the ones that make the structure of a Unity game object especially suitable to be addapted to an agent oriented paradigm.

Figure 2.1: Order of execution in Unity scripts



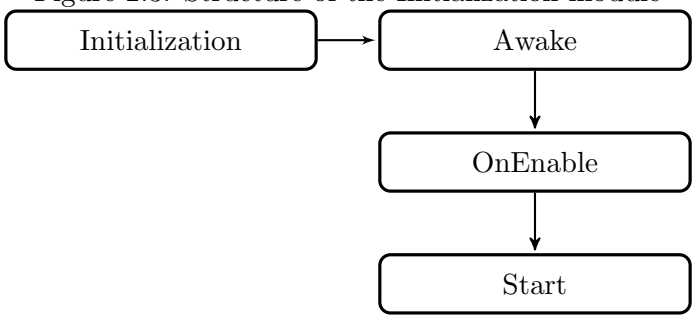
The editor is called when a script is attached or reset (Figure 2.2);

Figure 2.2: Structure of the Editor module



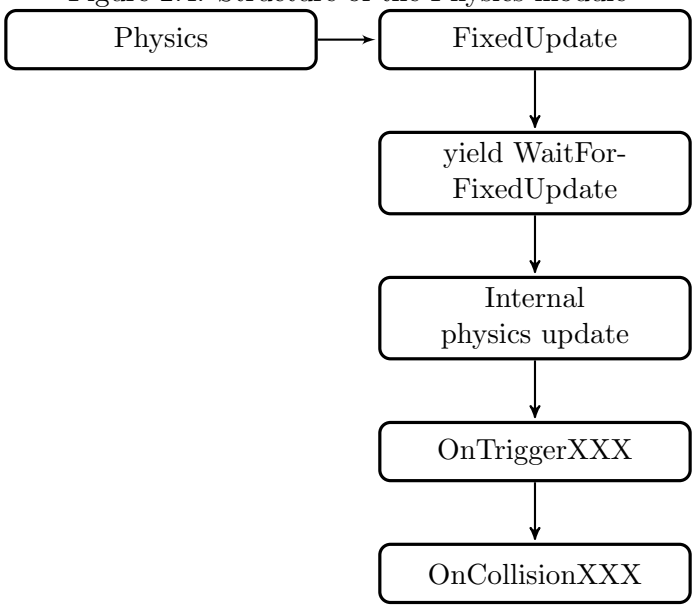
These functions are called only once for a given script (Figure 2.3), when an object is initialized.

Figure 2.3: Structure of the Initialization module



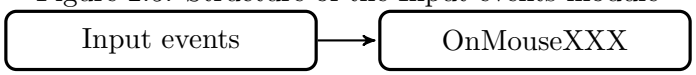
These functions can be called either once or multiple times per frame, depending on the internal update setting for FixedUpdate or events (OnTrigger, OnCollision) (Figure 2.4).

Figure 2.4: Structure of the Physics module



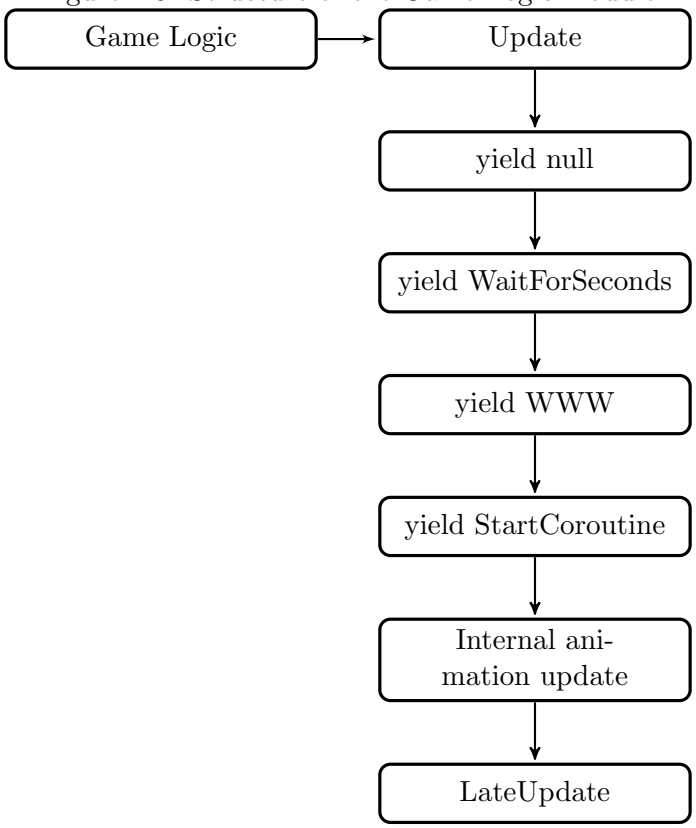
These routines are activated when mouse is clicked (Figure 2.5).

Figure 2.5: Structure of the Input events module



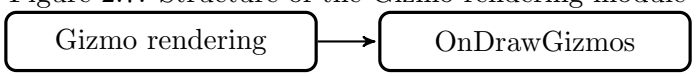
The routines in Figure 2.6 are called at fixed intervals. A function that has yielded previously resumes its execution during the update.

Figure 2.6: Structure of the Game Logic module



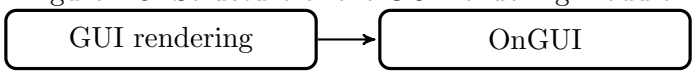
The Gizmo rendering module(Figure 2.7) It is used only when working in the editor.

Figure 2.7: Structure of the Gizmo rendering module



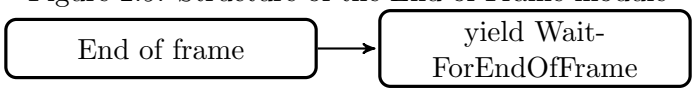
The GUI rendering module is called multiple times per frame update (Figure 2.8).

Figure 2.8: Structure of the GUI rendering module



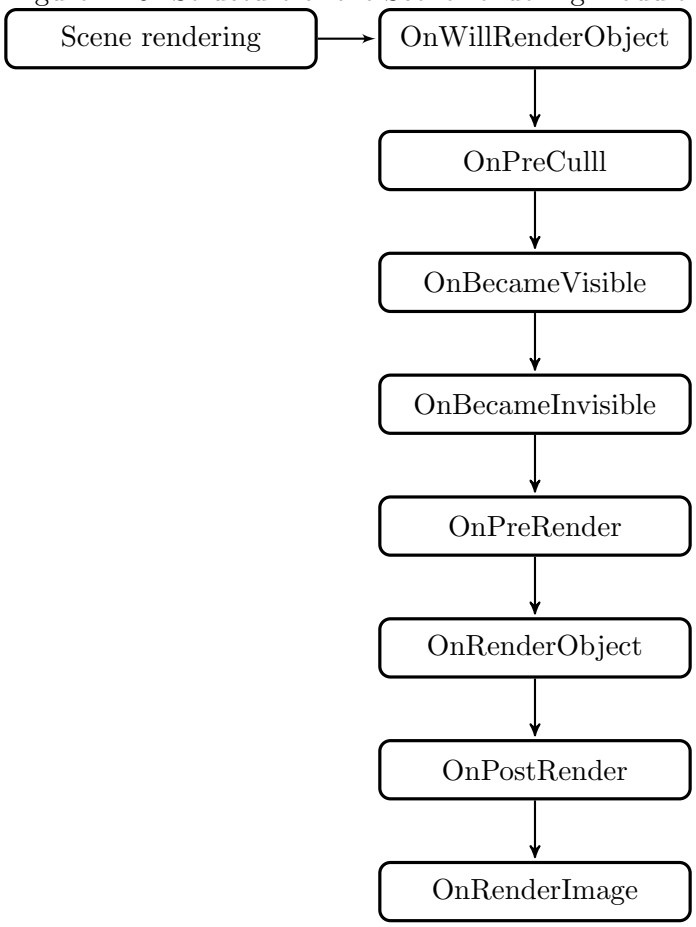
The End of Frame module is used when a frame ends (Figure 2.9).

Figure 2.9: Structure of the End of Frame module



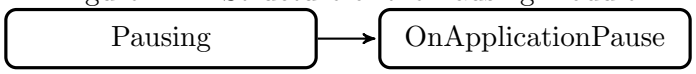
The Scene rendering module is called for each scene and each time rendering events happen (Figure 2.10).

Figure 2.10: Structure of the Scene rendering module



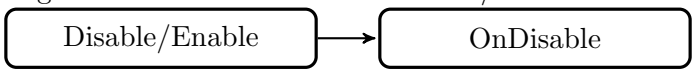
The pausing module is called when a frame has to pause, but another one has to be issued before the pause actually occurs (Figure 2.11).

Figure 2.11: Structure of the Pausing module



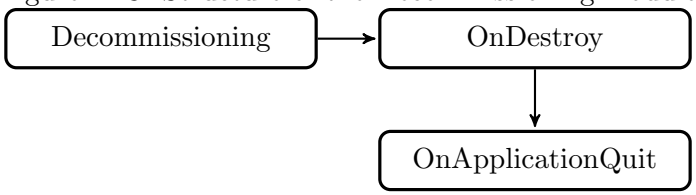
The Disable/Enable module is called when a script is disabled or enabled during a frame (Figure 2.12).

Figure 2.12: Structure of the Disable/Enable module



The Decommissioning modunle called when a script an object is no longer needed, irrespective of the frame (Figure 2.13).

Figure 2.13: Structure of the Decommissioning module



Chapter 3

MBTI software model

In this chapter a proposed model of implementing the MBTI personality classification will be examined. The model uses the succession of cognitive modes and the quantitative importance each of them has for each personality type as a way of predicting the behavior. This model will be tested against a list of job recommendations made for each of the types and aims to show that cognitive modes can be used to calculate how likely it would be for a candidate to be recommended each of the jobs from the list.

This section aims to show that the descriptive MBTI paradigm can be used also as a prospective model, from a computer programming point of view. The main goal of this step was to demonstrate that, although from a psychological point of view the MBTI classification is considered descriptive and not prescriptive, when it is implemented as an algorithm, the difference between proscriptiveness and descriptiveness is diminished. The results that were obtained in this section stand as a proof that the approach that has been chosen is a valid one and can be used as a basis for the more complex software solution that will be described in the next sections.

3.1 MBTI and job reccomendations

Considering that the MBTI was first introduced as a way to match workers to the best occupation, I decided to use occupations as a proof that the compatibility between an occupation and a personality type is dependent on the cognitive modes ordered as the archetypes from Table 2.3. The initial data that I found presented job suggestions for each of the personality types. The main goal was to be able to test that the qualitative psychological data (the MBTI classification, the cognitive modes, their degree of influence and their succession) for which the original data was created, could be adapted in a quantitative way (by assigning weights to each cognitive mode, based on its order for a MBTI type) and remain compatible with original data.

There are numerous career counselors that offer guidance based on the MBTI Type

characteristics and collected statistical data that counts how many people belonging to a certain type hold an occupation [19], [31], [1]. For my initial analysis I have chosen the career suggestions offered by [1]. The reasons for deciding for the list of suggestions presented here were:

- The number of jobs suggested per type was between 7 and 14, making the complete list long enough to be able to observe variations for the scores, but short enough so that it was easy to follow and interpret
- There were some occupations that were suggested to multiple personality types - this was particularly important, as it assured that all factors were taken into consideration when testing the hypothesis, as, in this case, it was expected that more than one MBTI type will have a high score for that occupation
- There was a wide range of occupations (e.g. professor, firefighter) - making it easier to understand the differences between scores when visualizing the results
- There were also jobs that require similar abilities and were listed separately (e.g. psychiatrist and psychologist)- in the end results, these professions were expected to have close scores between themselves, for most of the MBTI types

3.2 Personality Model

For this part of the project, a class has been implemented that contains all the necessary methods that can be used to quantify the MBTI types, their relation to the cognitive modes and how they can be used to deal with the list of career recommendations that was mentioned above. A UML diagram of this function can be seen in the figure below (Figure 3.1).

All the methods and variables of this class and their importance will be discussed in the next subsections.

3.2.1 Modes and personalities

The first step of creating a functional model is to be able to obtain the correct order in which the cognitive modes manifest for each of the personality types. The main methods that deal with that will be described below.

- The attitude (I/E; P/J) and function (T/F; N/S) letters are defined at the beginning of the class, using the class constructor without any parameters.
- All the 16 personalities are generated as a combination of 2 attitudes and 2 functions and saved in an array. This step is achieved using the method called CreatePersonalities().

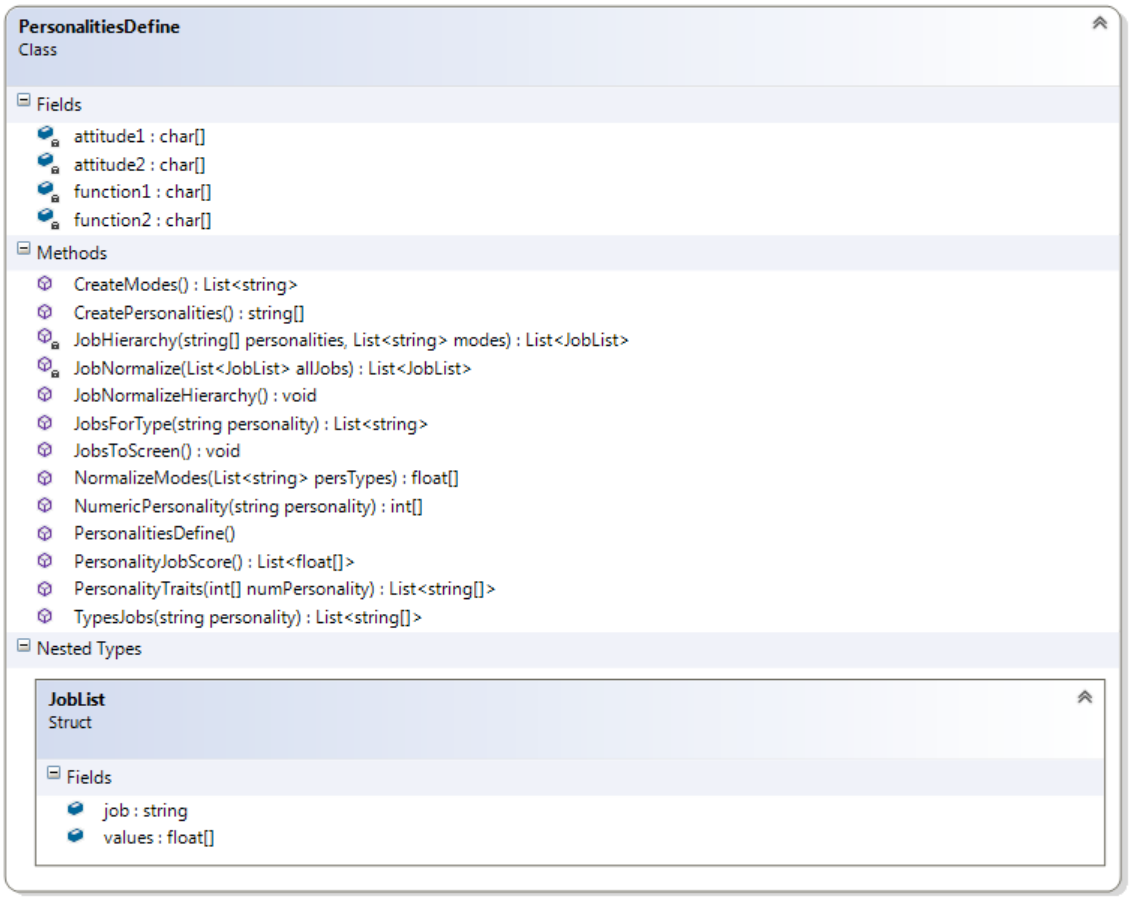


Figure 3.1: Structure of the PersonalitiesDefine Class

- Each of the personality types can either be represented using a string (e.g. INTP, ENFP, etc.), or in a numerical manner, by using the index corresponding to each of its attitudes and functions within the attitude and function arrays (e.g. INTP would be represented as 0000 and ENFP would be 1010). The NumericPersonality() method takes as a parameter the string representing a personality type and returns an array containing 4 integers that define that personality types using the order of its attitudes and cognitive functions.

- The cognitive modes are generated by combining the first attitude pair (I/E) with the function attributes by the CreateModes() method that is called without any parameters and returns an array containing all the 8 cognitive modes.

- The right order in which the cognitive modes manifest for a certain personality type is obtained using the rules that were stated in Table 2.3 of the Literature review section. The PersonalityTraits() method takes as a parameter a personality type expressed in its numerical form as an array of integers and returns a list in which the cognitive modes corresponding to that personality type appear in order.
  e.g. The INTP personality has the following cognitive mode ordering and scores assigned to each cognitive mode (Table 3.1):

Table 3.1: Cognitive modes' importance for INTP

Type	**Hero**	**Parent**	**Child**	**Anima**	**Opposer**	**Witch**	**Trickster**	**Daemon**
INTP	TI	NE	SI	FE	TE	NI	SE	FI
Score per mode	7	6	5	4	3	2	1	0

3.2.2 Job recommendations

The code deals with job recommendations that are stored in txt files corresponding to each of the personality types. The desired result is to get from a 4 letter personality centered recommendation (based on the MBTI type) to a cognitive mode centered recommendation, by analyzing each job type in connection with the order and importance of the cognitive modes within each of the personality types to which it was recommended to.

The following methods are used to adapt and process the raw job recommendations:

- The list of jobs that was recommended for a certain personality type is obtained using the JobsForType() method that takes a personality type in its string form and returns the jobs that were recommended for it as a list. The job lists are read from the text files were they were initially stored.

e.g The recommendations for ISFJ(Intraverted, Sensing, Feeling, Judging) are: Interior Decorators Designers Nurses Administrators and Managers Administrative Assistants Child Care / Early Childhood Development Social Work / Counselors Paralegals Clergy / Religious Workers Office Managers Shopkeepers Bookkeepers Home Economics

- The TypesJobs() method takes as a parameter a personality type in its string form and returns a list of arrays representing the succession of cognitive modes that is specific to that personality type. The list is correlated with the job recommendations that were made for that particular personality type.

- The job lists for all the personality types are then centralized in a single list and the scores that correspond to their cognitive mode lists are re-evaluated, in case the same recommendation is found to have been made to more than one personality type - each of the cognitive modes is assigned a value from 0 to 7 depending to its order of importance for the given personality type and if there is more than one type for which the same recommendation was made the values are added together (a complete example of how these scores are added and calculated can be seen in the Example subsection). The complete list of jobs and their corresponding scores for all the cognitive modes is obtained using the JobHierarchy() method that takes as parameters the complete list of personality types and a list containing all the cognitive modes.
  e.g. One and the same job has been recommended for an ISTP and an ISTJ, which have the following structure using the cognitive modes (Table 3.2):

Table 3.2: Example of defining the cognitive mode score

Type	**Hero**	**Parent**	**Child**	**Anima**	**Opposer**	**Witch**	**Trickster**	**Daemon**
ISTP	TI	SE	NI	FE	TE	SI	NE	FI
ISTJ	SI	TE	FI	NE	SE	TI	FE	NI
Score per mode	7	6	5	4	3	2	1	0

In this case, by adding up the scores that correspond to each of the modes, it can be concluded that this recommendation has a total score per mode as follows (Table 3.3):

Table 3.3: Example of calculating the initial cognitive mode weight for a job title

Mode	**TI**	**TE**	**FI**	**FE**	**NI**	**NE**	**SI**	**SE**
Total per mode	7+2=9	3+6=9	0+5=5	5+0=5	4+1=5	1+4=5	2+7=9	6+3=9

- The scores obtained above are then normalized so that they add up to 1 - this is done in order to decrease inaccuracies that might appear if there are disparities between the number of personality types that correspond to each job (e.g. if a certain job was recommended to 5 personality types then the score it has per cognitive mode might be extremely larger than that corresponding to a job that was only recommended to 1 personality type, and further calculations will be wrongly affected by this difference). The list of scores is processed through the JobNormalize() method.
  e.g. After the importance of each of the cognitive modes has been normalized such as their sum adds up to one, the results look as follows(Table 3.4), for the same example from Table 3.3 and 3.2:

Table 3.4: Example of calculating the normalized cognitive mode weight for a job title

Mode	**TI**	**TE**	**FI**	**FE**	**NI**	**NE**	**SI**	**SE**
Total per mode	0.16	0.16	0.09	0.09	0.09	0.09	0.16	0.16

To calculate the final scores that link the job recommendations to all the job types the following methods have been used:

- The importance that each of the modes has when determining the characteristics of a certain personality type are calibrated such as their sum adds up to 1 (so, instead of being integers that range from 0 to 7, they are all divided by 28, so they range from 0 to 0.25). This step is achieved by using the NormalizeModes() method.
  e.g. An INTP can be represented in the common cognitive modes succession as Table 3.5:

Table 3.5: Example of calculating the cognitive-mode normalized score

Mode	TI	TE	FI	FE	NI	NE	SI	SE
Weight of mode	7	3	0	4	2	6	5	1
Normalized weight of mode	0.25	0.11	0	0.14	0.07	0.21	0.18	0.04

- The initial score a job has with connection with a certain personality type is obtained by using the scalar product of the vector containing the scores that were calculated for each of the cognitive modes and the normalized strengths of the cognitive modes within that personality type. The PersonalityJobScore() method calculates these values.
  e.g. Continuing the exaple from the last paragraphs, the non- normalized score in this case would be 0.08 for INTP. The same job would obtain a non normalized score of 0.11 for an ESTJ personality and 0.12 for an ISTJ personality.

- The next step in to normalize the scores that are obtained by per job and personality type in the last step. This time, the interest is not to make them add up to one, but to scale them in an appropriate manner, so that they range in value from 0 to 1. the scaling is done for each of the personality types individually using:

$$NormalizedScore = \frac{InitialScore - minValue_{personalityType}}{maxValue_{personalityType} - minValue_{personalityType}} \tag{3.1}$$

  The results are then stored in separate files for each of the 16 MBTI personality types.

Each MBTI will have a list of suggestions that ranges in score from 0 (the last likely to be a good match) to one (the one that fits that personality type the best) - there can be multiple jobs that have a score of 1 or of 0 for any given personality type.

3.3 Results

After applying the methods described above to the complete list of job recommendations, a comprehensive list of scores was obtained (Appendix B).

When analyzing it can be observed:

- The jobs that were recommended for a certain personality type in the initial data obtain a very high score (above 0.85) - showing that the usage of cognitive modes correlates with the initial data.

- Personalities that are similar to each other obtain close scores for the same job type - this was expected, based on the fact that similar structures, in terms of cognitive mode succession should determine similar behaviors.
- Similar types of jobs obtain similar scores, irrespective of the personality type for which they are being analyzed - a result that validates the intuition that similar jobs would require similar psychological attributes (e.g. patience, analytical thinking, etc.).
- Most of the scores have values between 0.4-0.6 - signifying that there is no strong correlation between most job types and the MBTI types that have a different structure from the one for which they were recommended.

These results show that the approach that was used for adapting the MBTI psychological classification to a software model is valid and using the order of the cognitive modes and their weights is a valid solution to obtain realistic behavior.

3.4 Discussion

The experiment that was discussed in this section was used as a validation of the solution that will be proposed in the next sections of the thesis.

The fact that the obtained data matched the initial hypotheses and data proves that the MBTI personality classification model can be quantified and used for programming autonomous agents that exhibit different behaviors, based on their personality type, represented as a succession of cognitive modes that manifest with different strengths and frequencies.

Chapter 4

Agents as Objects

This section of the thesis contains a detailed explanation of how the BDI approach towards programming autonomous agents was adapted to object oriented programming (namely C#). The MBTI personality representation that was first discussed in the previous chapter was also included in the model. The elements that make up the software solution will be shown in UML diagrams when they are introduced .

The structure that is discussed here was implemented directly for the creation of autonomous game characters for the "Typology town" Unity game that was proposed as a application of all the elements discussed in the thesis. The specific information regarding the game can be found in the next chapter of the thesis.

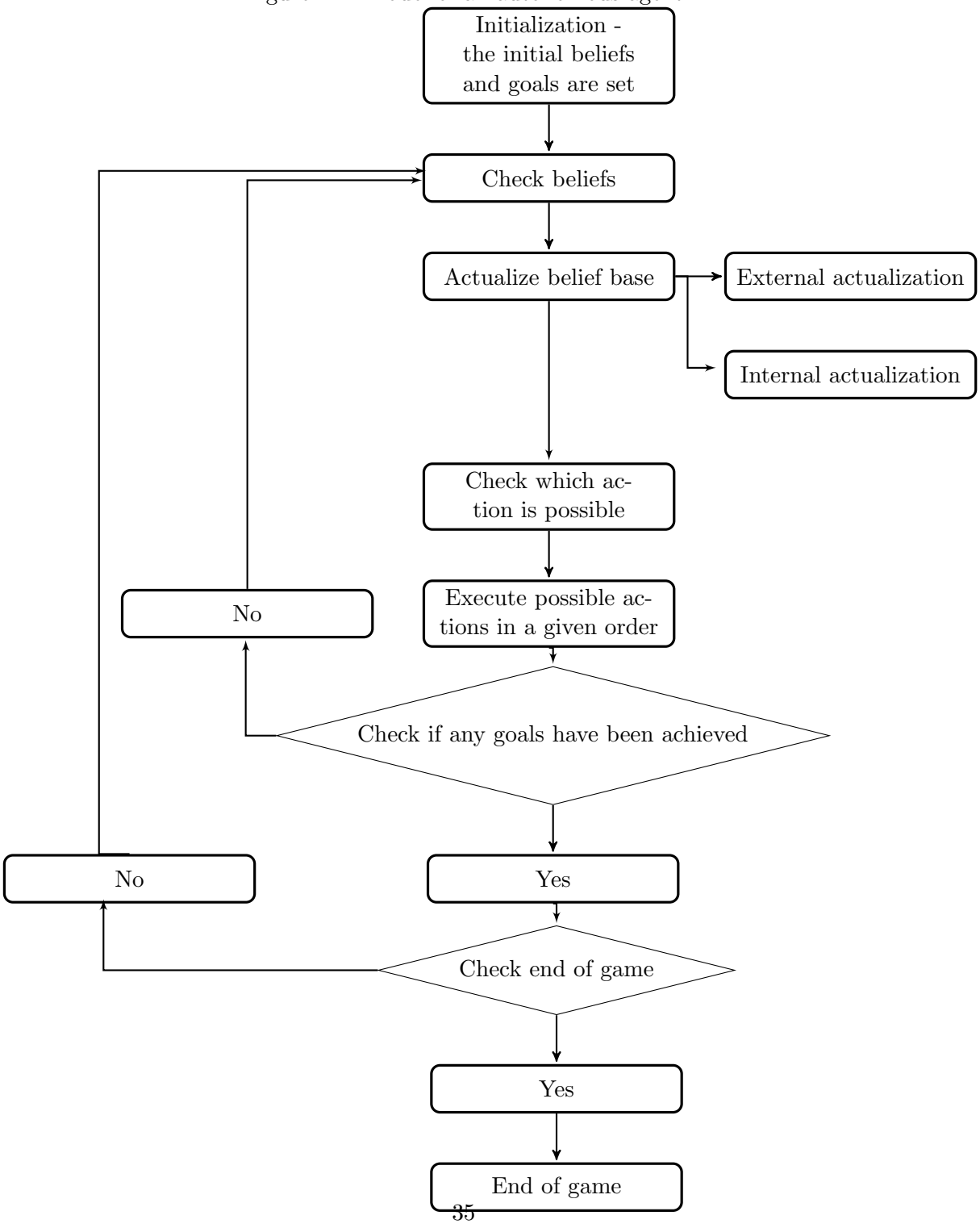
4.1 General Structure

The manner in which the autonomy of an agent is defined using BDI concepts is through beliefs, goals and plans. These concepts can be adapted using software objects that can be used directly in the Unity programming environment, through a structure as shown in Figure 4.1.

The elements presented in the figure can be defined as follows:

- The belief base - is the complete list of beliefs an agent holds at a given time. The beliefs held at a moment cannot be conflicting (e.g. it cannot believe that it is both alive and dead).
- Actualization of beliefs - this is done in two manners - by methods that are called by the agent itself (called Internal actualization in Figure 4.1 and executed at fixed intervals -through the Update method in Unity - as presented in Figure 2.6, for example) or through methods that are called by other agents or elements of the environment (called External actualization in Figure 4.1 - when certain events happen or game elements appear in the scene).

Figure 4.1: Model of an autonomous agent



- Plans - are represented as successions of individual actions that have a list of necessary beliefs that need to be true for the action to be executable. The list of plans and conditions that need to be valid is defined at the construction of the agent. Each of the actions that composes a plan has its own list of necessary preconditions. While an action is being executed, it is possible that the belief base of the agent is modified.
- Goals - a goal is accomplished when a list of conditions hold true. In this particular implementation, goals will be represented as ordinary actions that the agent can execute if their preconditions are valid. The achievement of certain goals can lead to the ending of the game,while others may only result in a change in the belief base of the agent and might pass as unnoticed by an external observer.

4.2 Beliefs and Plans

The way in which beliefs and plans of the autonomous agents were implemented will be discussed firstly in a general manner, in terms of how the BDI paradigm was adapted to the C# / Unity object oriented programming environment. The next section puts more emphasis on how the MBTI personalities have been represented in the proposed software solution.

4.2.1 Beliefs

In this proposed model, the beliefs of the agent are represented by two data structures:

- The complete belief base - contains a list of types of beliefs and all the possible values these beliefs can hold during the whole execution; an agent can only hold one value of each type at any given time. An example of categories of beliefs and their possible values is shown below.

  Belief type: Hungry ; Values: Yes No;

  Belief type: Friendliness; Values: Friend Neutral Enemy;

  These values are set at the beginning of the execution and may not be modified. They are used as a guideline for determining all possible belief states that the agent might hold.

- Current beliefs - These are the beliefs that the agent holds at a given moment. They consist of a belief of each of the types stated in the complete belief base. An example of the current beliefs an agent holds at a given moment, using the same complete belief base stated above is:

Belief type: Hungry ; CurrentValue: Yes;

Belief type: Friendliness; CurrentValue: Neutral;

- Belief data structures (both current beliefs and complete belief bases) are organized in two categories based on their utility in the agent's behavior - general beliefs and beliefs that are only relevant for behaviors associated to a certain cognitive mode; this is done for optimizing the speed of access to a certain belief type during the execution of the code (for testing the truth value or modifying it).
- For easiness of access and modification, the beliefs are written in .txt files that are read when the code is executed.

4.2.2 Plans

The proposed solution adapts the concept of plans to object oriented programming as follows:

- A plan consists of the names of methods that should be called by the agent followed by a list of preconditions that need to be true for the action to be executed.
- The list of post-conditions after a plan is executed is not explicitly specified. It is possible that, within the actions that are executed by the called method, some beliefs of the agent are modified, but there are also plans that do not affect the current beliefs of the agent.
- The preconditions for each of the plans are similar in structure to the current beliefs (one belief belonging to each type).
- Plans are also organized in general plans and plans that call for actions specific to one of the cognitive modes. The preconditions of a plan are also determined by its connection with a cognitive mode (e.g. a plan that is listed under the TI cognitive mode will have preconditions that are listed as TI beliefs).

4.2.3 Goals

In this solution goals are not represented separately from plans and beliefs, instead, they are represented implicitly and show the following characteristics:

- Goals, in the way in which they are represented in the BDI paradigm, appear here as preconditions for a special type of plan.
- A goal is reached (in this case, when a list of conditions represented as beliefs that the agent needs to have are equal to the current beliefs of the agent) when the plan that had the same precondition as the goal is executed.

- After a goal is reached in this application, it becomes visible that it had happened - the score is updated, the game is over, etc.
- Using this representation, the goals are not stated separately, but rather implemented directly in the software solution and tested through the same pre-condition rules used for plans.

4.3 Integrating personalities

In this section, the MBTI types' implementation that was described in the first two sections will be introduced to the model. The personality types will impact the behavior of an agent using the influence of cognitive modes, in a manner that is similar to the one that was proposed and tested in Chapter 3 of the thesis.

4.3.1 Personality Class

Similar to the preliminary model proposed in Chapter 3, an agent's personality needs to be defined as a succession of cognitive modes for flexibility in determining behaviors that can be easily adapted to each of the agents. The flexibility is assured by the fact that complex behaviors are determined by simple parts that are declared in a modular manner, and their succession and frequency of manifestation play a key role.

Some of the desired ussages of the methods declared in this class are :

- For each of the personality types expressed in a 4 letter format the list of cognitive modes ordered by their importance can be obtained;
- The cognitive modes are attached to the 8 archetypes, as presented in Chapter 2;
- The personality is specific to each of the agents present in the game, creating diverse behaviors.

Figure 4.2 shows the general structure of the Personality class that was used in this part of the project.

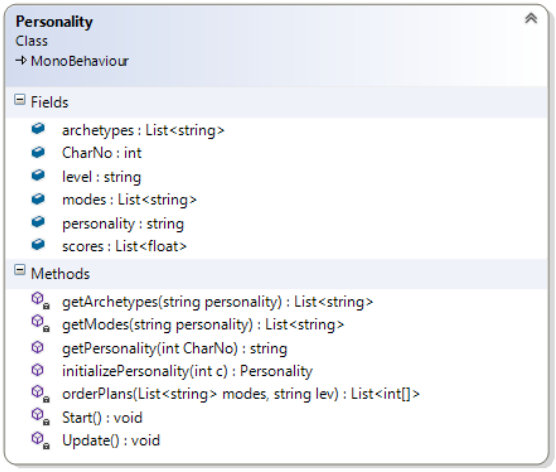


Figure 4.2: UML of the Personality class

It can be seen from this UML that the Personality class used here contains variables and routines that were described in the previous chapter - regarding archetypes, cognitive modes and personality types, alongside with Unity specific routines- start, update and routines that link the cognitive modes succession to the execution of plans.

The list of the cognitive modes that is specific to each personality type is used to determine the behavior of the agent through the order and frequency with which he will try to act upon his plans (an agent will try to act on plans that belong to a cognitive mode that is more important for its MBTI types more times than it would for ones that belong to less important cognitive modes).

4.3.2 Beliefs, Plans & Goals

When introducing elements that are specific to the personality types on the structure of the representation of Beliefs and Plans (as it was explained in the previous section, Goals are not explicitly and separately stated) cognitive-mode related beliefs and plans are created.

The Beliefs and Plans are represented as arrays of 9 elements each, where every element is a data structure of type Belief and Plan, respectively. The structure of the arrays is fixed - the first position contains the general beliefs/ list of plans and the next ones are occupied by the beliefs and plans that are specific to each of the 8 cognitive modes. The cognitive modes appear in the order of their importance for determining the behavior of the personality type belonging to that agent.

As an example, the order of cognitive modes for an ESTJ (Extraverted, Sensing, Thinking, Judging) personaliity is: TE (Extraverted Thinking), SI (Intraverted Sensing),NE (Extraverted iNtuition), FI(Intraverted Feeling), TI (Intraverted Thinking), SE (Extraverted Sensing), NI (Intraverted iNtuition), FE (Extraverted Feeling). For this MBTI type, the

plans and beliefs will be organized in this order: General Beliefs & Plans, TE specific Beliefs & Plans, SI specific Beliefs & Plans, NE specific Beliefs & Plans, FI specific Beliefs & Plans, TI specific Beliefs & Plans, SE specific Beliefs & Plans, NI specific Beliefs & Plans, FE specific Beliefs & Plans.

4.3.3 Activities

The plans of an agent consist of successions of activities. As described in the previous subsections, plans can be either general or cognitive mode specific. The same can be said about the activities that are part of the plans. Multiple Activities classes were implemented, one per each cognitive mode. All of them are derived from a common one, that handles actions that are not allocated to any of the cognitive modes. Figure 4.3 shows a UML of the main Activities class and all its methods, as they were implemented in the game.

These Activities methods are called using the agent's plans, depending on the cognitive mode through which the agent is acting at that certain moment and their general and cognitive mode related current beliefs. The number of calls made to each of the methods is dependent on the cognitive mode to which it is attached - the higher the importance of the cognitive mode within the agent's personality, the more tries of executing a plan the agent will perform (the number of tries is equal to the weight of the cognitive mode for each MBTI type - as described in Chapter 3).

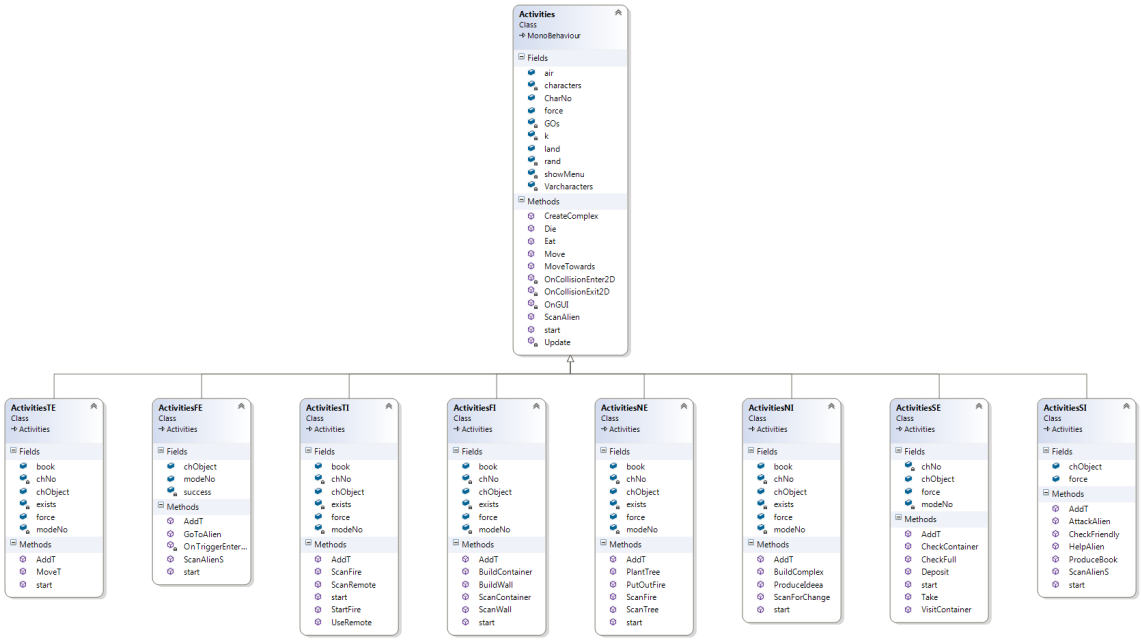


Figure 4.3: UML of the Activities class

4.4 Autonomy

The Autonomy class is the part of the code that links together all the other classes and methods and assures that the agent behaves autonomously and is able to adapt to its environment and pursue its goals, while also taking into consideration the effect that personality types and cognitive modes have.

Figure 4.4 shows the UML of the Autonomy, Beliefs and Plans classes.

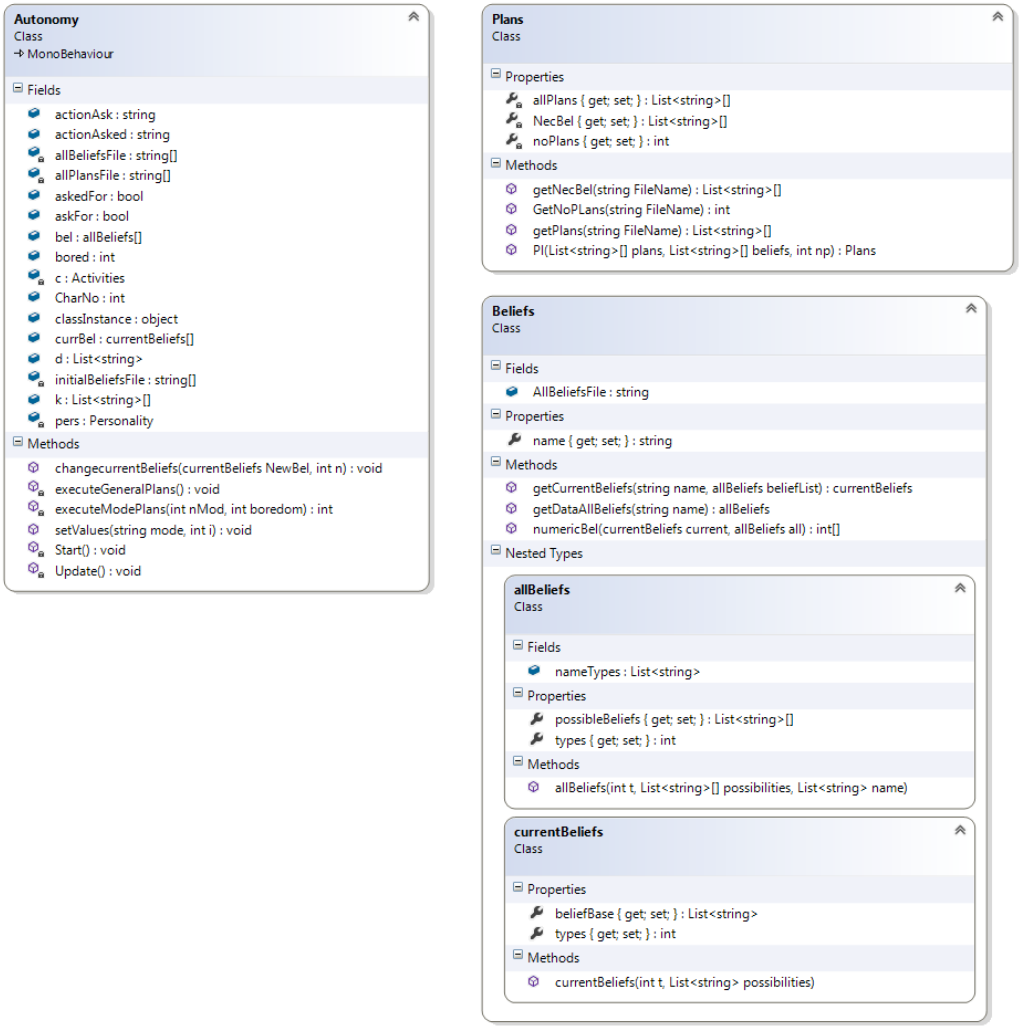
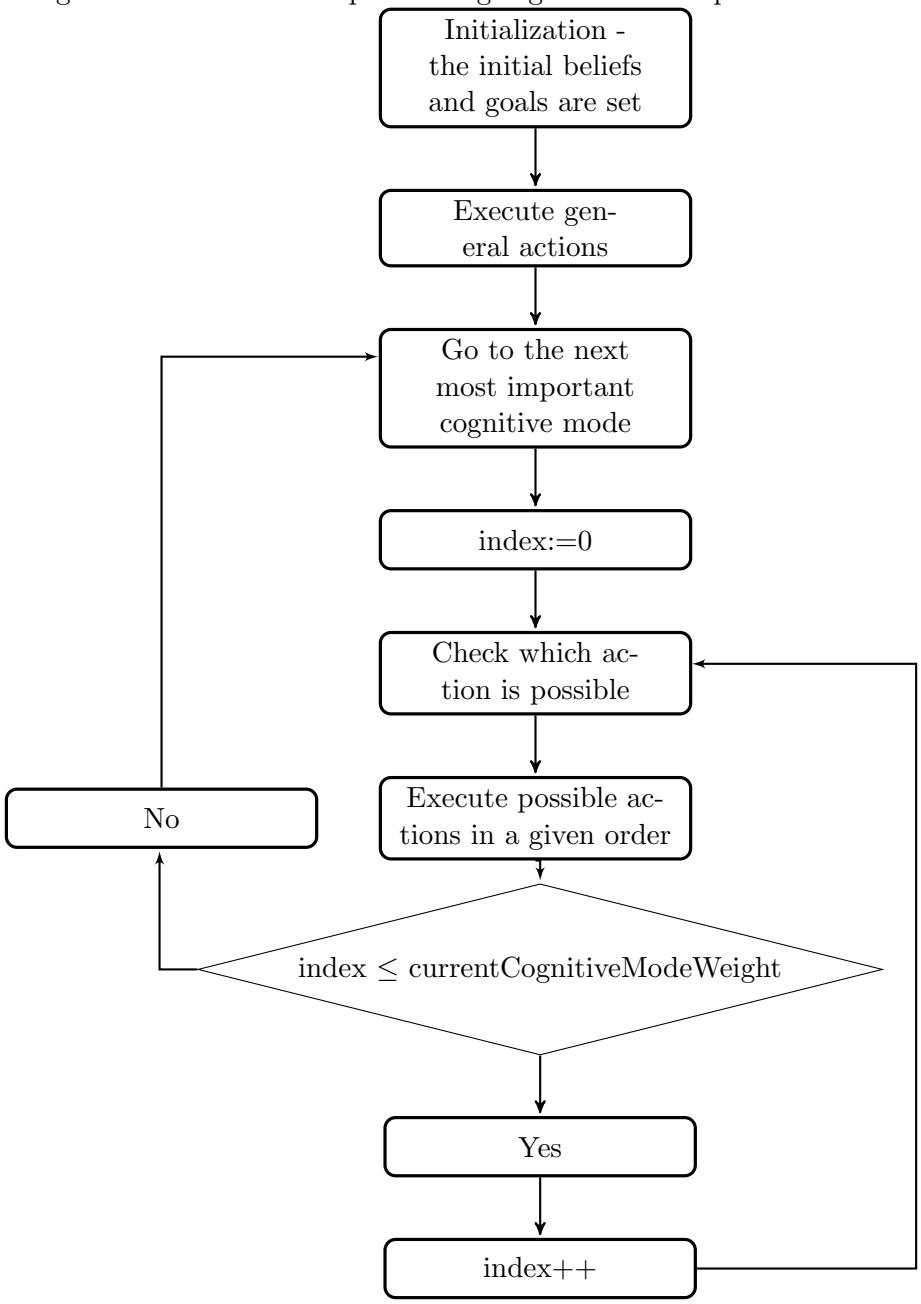


Figure 4.4: UML of the agent autonomy classes

The way in which an autonomous agent that has an MBTI personality is represented schematically in Figure 4.5.

Figure 4.5: Execution of plans using cognitive mode specific actions



The execution of plans by an agent is done in cycles, meaning that an agent will try to perform all possible actions in a continuous cyclical manner until it becomes impossible to do any action or the goal of winning the game has been fulfilled.

As it can be seen in Figure 4.5, a cycle consists of one try to execute plans belonging to the general category, followed by cognitive mode specific plans that are executed for a number of times that corresponds to their impact in determining the behavior of a certain personality type (the number of loops in a cycle was implemented as a representation of the weights attached to the cognitive modes for each MBTI type in the previous chapter).

The validity of this approach has been proven in Chapter 3, where personality types were decomposed in their cognitive modes and each cognitive mode was assigned a weight that signified its importance and impact upon the behavior of the agent (in that case, the suitability of a person to perform a certain job or job type).

In this case, the weights are represented by the number of executions of each cognitive mode related plan loop - from 8 executions for the cognitive mode that impacts the behavior the most to 1 execution for the least important mode.

Figure 4.5 can be seen as an extension of Figure 4.1 (actualizing beliefs, checking pre-conditions,execution of plans), that explains how the cognitive modes are integrated in the model.

4.5 Results and Discussion

The solution that was proposed in this section shows that it is possible for agent oriented programming paradigms to be adapted to a object oriented programming environment. The MBTI personality types were also integrated in the solution, using the tools that were tested in Chapter 3 of the thesis.

The software solution that was described here was ported to the Unity programming environment were it was used for creating game characters that act like autonomous agents and exhibit characteristics that are specific to the MBTI personality types.

The desired characteristics for this solution that were achieved can be stated as follows:

- A software solution that assures that the main elements of the BDI and agent oriented programming are represented efficiently;
- A clear class structure that makes the solution as flexible and portable as possible, by dividing plans, beliefs and actions using the cognitive modes;
- Being able to work in a multi agent environment;
- A clear way to produce different behaviors for agents that were assigned different personality types, while still responding to elements in the environment in a useful manner;

- Adaptation of the cognitive mode weights as a means of determining the personalities, similar to the way described and tested in Chapter 3 of the thesis.

The next section will show how this solution works in a computer game where a human player can observe and interact with autonomous game characters whose behaviors are determined using the algorithm and data structures presented in this chapter.

Chapter 5

GamePlay

The game that is proposed as an illustration of how the agents behave is an action game, more precisely a platform game that consists of a playable character and multiple autonomous ones that interact with the player, by either helping him to reach his goal or trying to stop him from that.

The following sections will describe the story line of the game, some details about the game play and more general considerations about the project as a whole.

5.1 Story line and goals

5.1.1 The story line

An alien from a far away galaxy has emergency landed on earth due to a technical difficulty with his spaceship. Unfortunately the damage to the ship was big enough that he will need to get some earth materials in order to fix it and has to interact with the humans to be able to get them. Although the alien's primary goal is to be able to fix his spaceship in time, he gets to learn things about what makes humans act in certain manners, depending on their MBTI personality types - and he can get to create a better plan for gathering resources if he can behave in a way that considers the humans' probable strategy.

The behavior of an agent that acts according to a mode is defined based on the short description of the cognitive modes offered below (Table 2.2 from the Chapter 2 describing what MBTI personalities are and how they manifest through the cognitive modes).

Table 5.1: Cognitive modes description

Cognitive mode	Essential value	Description
SE	Experiment	Discovers new ideas and phenomena by direct experience
NE	Ideation	Rearranges known concepts into novel systems
TE	Organization	Efficiently manages resources, decisive, imposes structure
FE	Community	Expressive, tactful builder of group morale
SI	Knowledge	Physically self aware, values practice and technique
NI	Imagination	Prophetic, guided by inner fantasies and visions
TI	Analysis	Logically improves rational performance
FI	Evaluation	Uses personal values to distinguish good from bad

5.1.2 Goals

The main goal for the player is to collect all the objects necessary in order to fix the spaceship before its life is over.

The human characters (non-playable characters) inside the game are also trying to collect the objects that appear in the game space. If the characters manage to collect all the objects before the alien does, or if the alien (player-controlled character) runs out of energy, the game is lost.

The player wins the game if they manage to bring to the spaceship an object of each type (simple and complex objects alike).

5.1.3 Rules and structure

"Typology town" is a platform game (a game which involves guiding an avatar to jump between suspended platforms, over obstacles, or both to advance) that shows an average of 5 platforms per frame. The characters (both user controlled and autonomous) evolve mostly on the platforms, but there is also a ground level where the spaceship has crushed.

All the primary objects (described in the next section) are to be found on the platforms.

The buildable objects can be placed anywhere in the gamespace - platforms or ground level.

The characters can always fall or jump from the platforms to the ground without any immediate damage but they have to get to certain location where a platform is low enough for them to return to the "higher" game-space.

The spaceship wreck is set at a fixed location, at the beginning of the game-space.The alien has to travel to it each time it has gathered enough objects to make a fixture.

The interactions between the characters and the alien depend on the 'Dissatisfaction' score that each of the characters has. At the beginning of the game this score is equal to 40,

meaning that the character will behave cooperatively towards the alien. The score increases or decreases based on the game development (a complete list of how scores increase and decrease can be seen in Table 5.5 of section 5.3), and a score over 50 will determine the character to behave negatively towards the alien (e.g. shoot it when in proximity).

The characters can use a 'Container' after it is built to deposit objects they have collected. The objects that were deposited in a container remain there even after one of the characters has been killed.

The player controlled alien navigates through the game space using the arrow keys for left and right movement, the space bar for jumping and the "S" key for shooting. The objects that the alien posses can be seen by accessing the menus that are present in the game. Complex objects can also be build using the menus. The alien can deposit any number of objects in the spaceship, but can only hold 10 objects in immediate possession, unless it has accomplished a quest, and then the number increases temporarily to account for all the objects that were gained during the quest.

5.2 Character Behavior

The characters act according to the MBTI personality types. Each of the types is manifested by all of the 8 cognitive modes in a given order. All of the modes have a characteristic primary object as shown in Table 2 and a set of actions that the agent performs if they are possible.

Table 5.2: Objects and movement patterns that are connected to the cognitive modes

Cognitive mode	**Essential value**	**Object**	**Activity**
SE	Experiment	Backpack	Deposits or takes objects from a container
NE	Ideation	Idea	Plants a tree or puts out a fire
TE	Organization	Ladder	Walk towards current most needed object
FE	Community	Telephone	Goes towards the alien
SI	Knowledge	Book	Attacks or helps the alien
NI	Imagination	ColourGenie	Builds a complex object out of objects it possesses
TI	Analysis	Plan	Starts a fire and moves the container object around the game space
FI	Evaluation	Heart	Builds a wall or creates a container

A character will always add the object that represents the cognitive mode that is manifesting at a given moment to its 'Desired Objects'(the list of objects that it can collect at a given time if it touches them) list if it does not posses it. It will only add other objects if its behavior specifically asks for them (e.g. it is trying to build a complex item for which they are needed).

The next table (Table 5.3) describes the relationship types that appear between the characters and the alien during the play of the game.

Table 5.3: Relationship types

Relationship	**Cause**	**Effect**
Friendly	Dissatisfaction ≤ 50	Cooperate
Hostile	Dissatisfaction > 50	Attack

The life index of a character may also affect its behavior as shown in Table 5.4:

Table 5.4: Life index

Life	**Cause**	**Effect**
Hungry	LifeIndex ≤ 50	Go towards food item
OK	LifeIndex > 50	Continue regular actions

5.3 Game Items

Apart from the objects that are characteristic to each mode, there are general objects that can be collected by the autonomous characters or the alien when needed. They can have an immediate effect or they can be stored and used later to create complex objects.

All complex items are made of three components that can be either mode-objects as described in Table 5.2, other complex objects or simple objects. Each complex object has at least one component that is another complex object or a mode-object.

Table 5.5: Complex objects

Item	Pre-requisites	Relationship	Effect
Fire	Match, Wood, ColourGenie	Enemy	If touched provokes -20 life points
Container	Book, Brick, Ladder	Friendly	Stores objects, they can be retrieved at any time when needed
Tree	Heart, Water, Wood	Friendly/Alien	Gives an apple any time it is watered. The apple appears in a 5 platform range of the tree. The dissatisfaction index decreases by 10.
Axe	Tree, Book, Ideea	Enemy/Alien	Used to cut down a tree Produces 5 wood objects scattered around 10 platforms. The dissatisfaction index grows 10 points.
MovingPlatform	Ideea,Wood,Plan	All	Can only be used on the ground. Used to move the container along the game-space.
Wall	Brick, Ladder, Backpack	All	No character can go past it directly without jumping on a different platform or going to the ground level.
WaterCan	Water, Backpack, Plan	Friend/Alien	Used for watering the trees -1 dissatisfaction or for puting off a fire.
RemoteControll	Telephone, Heart , Wire	All	Used to move the platform to a new location from a distance
Hammer	ColourGenie, Telephone, Wood	All	Destroys walls +3 Dissatisfaction
Match	Wood, Brick, Hammer	All	No direct effect. disappears after one use.
Gun	Match, Wire, Hammer	Enemy	No effect.

Table 5.6: Primary non-mode objects

Item	Effect and Production Method
Wood	It's both an initial item and can be produced by cutting down trees
Brick	Only appears as an initial item.
Water	Only appears as an initial item.
Wire	Only appears as an initial item.
Apple	Produced by Trees when watered

5.4 Other specifications

If a person character dies, it disappears, but two extra villagers come to the game-space. The two new extra characters start with an 'hostile' relationship mood towards the alien and act accordingly. This occurs every time an autonomous character dies, irrespective of the number of humans(non-playable characters) already present in the game.

The characters and the alien can move at the same speed.

No character can posses a number of items that is bigger than 11, but if an object is stored in a container it is not considered as a possession.

Some screenshots of the game can be seen in Appendix C of the Thesis.

5.5 In game Menus

During the game there are three kinds of menus that appear in the GUI:

- Character Stats menu - shows the energy levels for each of the characters and their dissatisfaction score, alongside with the ordered cognitive modes for the personality of each of the present characters and the number of times a mode characteristic object(Table 5.2) was added by a character to its 'Desired Objects' list in the game.
- Alien Options menu - includes the option to view and/or use the collected items, the possibility to chose to go on a quest and the choice of building complex objects using the objects that it already has in its inventory.
- Other GUI - these include:
  - SpaceShip Menu - it appears when the alien is in proximity to the space ship wreck and offers the option to deposit an object in the space ship or take an object from the ship inventory and deposit in the alien inventory

– Personality Choice for new character - appears after a character has been killed and makes the player chose the personality type that the new character that will be created will have.

5.6 Quest

The game offers the option for the player to go on a "Quest". In this case the player will have to interact more with the autonomous game characters.

This option was introduced for gamification purposes, as it becomes more entertaining if the player becomes more involved with the game environment and has to chose explicitly if he wants to take a risk.

If the GoOnAQuest option is chosen, the player will then have to select the character that he wants to find in the game space. After the choice is made, a timer is started and the alien needs to find and collide with a character that has the desired personality type. If this character is found before the time runs out, and if in the meantime no other character has been touched, the alien automatically takes all of this character's resources and all the resources that were deposited in the container. In this case, the character will also lose 50 energy points and its dissatisfaction score will increase by 20.

If the time runs out before the right character is found, or if the alien touches another character first, the alien will lose all of its objects, alongside with the resources that were deposited in the spaceship. The alien will also lose 20 energy points.

5.7 Multi-agent Considerations

The game starts with a single player controlled character(the alien) and a single autonomous agent. As the game progresses, the number of autonomous characters may increase (as it was stated above, every time a character is killed, two others appear), making it a multi-agent application.

At this stage of the development, this aspect has not been fully explored in terms of implementing a protocol of communication between agents or establish a way in which they will compete for resources (e.g. 2 agents are trying to get a single object that they both need) and these are matters that will be included in possible future developments of the project.

The multi agent aspect of the project at this stage consists of the fact that the autonomous characters collaborate with each other, because they share the attitude towards the player controlled character - so they will all help or try to kill the character.

Another important characteristic of the multi agent system that can be observed here is that the characters can share resources towards a common goal - when a container is build, all agents are able to deposit objects in it or build complex objects using the objects that were deposited by others.

5.8 Results and Discussion

'Typology Town' was used as an illustration that the concepts expressed in the past two chapters can be successfully implemented. By observing and interacting with the game characters it can be observed that behavior is different, based on the MBTI type that was assigned to them and based on other elements that are present in the game environment.

By playing the game, it can be observed that the weights that the cognitive modes have in determining each psychological type are observable by the actions that characters chose to perform:

- Strong Fe types will constantly follow the alien;
- Strong Te types will tend to collect more objects more quickly;
- If a character manifests a stronger Ne, Ni or Fi preference, then it will tend to build the objects that are related to these types in Table 5.2 more quickly than other types
- The strategies of the player can be adapted to the personality types that were assigned to the characters.

The differences between the behaviors of the characters would be greater if the list of actions assigned to each cognitive mode was larger, but the fact that it is noticeable in this game is strong evidence that the proposed algorithm works for creating diverse human-like behavior for autonomous characters.

Chapter 6

Conclusion

The last three chapters have introduced an algorithm that adapts the MBTI personality types to a structure that is compatible with object oriented programming, expressed a way in which this approach can be used alongside with an adaptation of BDI specific constructs (beliefs, goals and plans) and then described how these concepts were used together with elements of game design and gamification to create a computer game that contains multiple autonomous characters and a user controlled one.

The concluding remarks and possible future developments in this final chapter of the thesis can be structured according to the three categories stated in the past chapters: personality theory, agent programming and game development.

6.1 Psychological types modelling

6.1.1 Conclusion

From the point of view of modelling the personality types according to the MBTI classification, this thesis has highlighted that:

- The 16 personality types are defined by the succession and impact of the 8 cognitive modes.
- The cognitive modes can be analyzed as independent factors from one another.
- By using the succession and attaching weights to the cognitive modes, it is possible to create a software model that accounts for the way in which belonging to a personality type affects the behavior and choices of a person (this model only accounts for tendencies and is a simplification, not accounting for many other psychological factors).
- There are other personality theories and classifications that could be used independently or in correlation with the typology introduced by the MBTI.

6.1.2 Future Developments

The model introduced in this thesis does not account for many other factors that might play a role in determining the real behavior of somebody, based on psychological theories. Some aspects that were not fully explored in this thesis but could be researched in the future consist of:

- The archetype roles could be introduced in the model, as a further quantifier in the model.
- The concept of positive and negative archetypes can be added to the model.
- Other personality models (the Enneagram and The big five) can be integrated or compared with this model to create a better representation of the factors that determine human behavior and personality.

6.2 Agents and Objects

6.2.1 Conclusion

From the part of the project that aimed at creating a framework that made concepts that were specific to the agent oriented programming environments (3APL) usable in a object oriented programming language (C#), it can be concluded that:

- Beliefs, Plans and Goals can be represented using software objects, although their implementation requires defining extra structures, and more programming effort overall.
- The structures that are created for this purpose can be easily ported and adapted to be used in other object oriented environments - the portability of software objects is higher than the one that exists for solutions written in agent oriented programming languages.
- The agent oriented programming languages offer a clearer representation of the BDI logic concepts that are used in programming autonomous agents, but the same functionalities can be achieved by using objects.

6.2.2 Future Developments

The framework that was created within this project assures that the basic elements of agent oriented programming were brought to use in a C#/ Unity environment, but there are some ways in which the project could be extended in the future:

- The solution already offers limited support for agent messaging and other multi-agent aspects, but there are still many aspects that could be optimized or introduced in this pursuit.

- In this implementation, Goals are not represented as an independent software class, and while this approach worked for the current project, improvements could be made to cover a broader range of applications.
- As the number of beliefs and plans increase, solutions need to be fast and optimizations could be considered, in order to make them work faster when the dimensions of the data they work with increases.

6.3 Game Development

6.3.1 Conclusion

'Typology town' was created to showcase the concepts that were introduced in the past two sections. By playing the game it can be observed that:

- The two parts of the solution can work together and form a singular software project.
- The Unity programming environment is a useful tool for creating computer games, making it easier for the programmers to concentrate on implementing behaviors and actions for their characters, as game physics and other game development elements are preimplemented.
- The way in which the personality types have an impact upon the autonomous characters' behavior can be observed in an interactive, gamified manner.

6.3.2 Future Developments

From the point of view of game development, some future directions for improvement consist of:

- Introduction of more actions for each cognitive mode, so that the differences in behavior become more obvious.
- Introduction of new levels where characters collaborate more with each other.
- Improvements of the gamification aspects by creating more interactions between the user controlled character and the autonomous ones.

Appendix A

Cognitive modes and archetypes attached to the MBTI Types

Table A.1: Types and Archetypes

Type	**Hero**	**Parent**	**Child**	**Anima**	**Opposer**	**Witch**	**Trickster**	**Daemon**
ISTP	TI	SE	NI	FE	TE	SI	NE	FI
ISTJ	SI	TE	FI	NE	SE	TI	FE	NI
ISFP	FI	SE	NI	TE	FE	SI	NE	TI
ISFJ	SI	FE	TI	NE	SE	FI	TE	NI
INTP	TI	NE	SI	FE	TE	NI	SE	FI
INTJ	NI	TE	FI	SE	NE	TI	FE	SI
INFP	FI	NE	SI	TE	FE	NI	SE	TI
INFJ	NI	FE	TI	SE	NE	FI	TE	SI
ESTP	SE	TI	FE	NI	SI	TE	FI	NE
ESTJ	TE	SI	NE	FI	TI	SE	NI	FE
ESFP	SE	FI	TE	NI	SI	FE	TI	NE
ESFJ	FE	SI	NE	TI	FI	SE	NI	TE
ENTP	NE	TI	FE	SI	NI	TE	FI	SE
ENTJ	TE	NI	SE	FI	TI	NE	SI	FE
ENFP	NE	FI	TE	SI	NI	FE	TI	SE
ENFJ	FE	NI	SE	TI	FI	NE	SI	TE

Appendix B

Scores for initial job matching

In the following table the normalized scores that were obtained by all the personalities when considering the complete list of occupations can be seen in the tables below:

Table B.1: The extroverted Types

Job Name	**ENFJ**	**ENFP**	**ENTJ**	**ENTP**	**ESFJ**	**ESTJ**	**ESFP**	**ESTP**
Accountants	0.32	0.63	0.25	0.64	0.75	0.68	0.36	0.37
Actors	0.35	0.80	0.35	0.84	0.65	0.65	0.16	0.20
AdministrativeAssistants	0.55	0.45	0.02	0.82	0.98	0.45	0.18	0.55
Administrators	0.60	0.50	0.00	0.85	1.00	0.40	0.15	0.50
AlternativeHealthCarePr.	0.95	0.24	0.50	0.63	0.50	0.05	0.38	0.76
Artists	0.56	0.54	0.64	0.32	0.36	0.44	0.68	0.46
Athletes	0.79	0.02	0.55	0.48	0.45	0.21	0.52	0.98
Attorneys	0.45	0.55	0.60	0.62	0.40	0.55	0.38	0.45
Bookkeepers	0.55	0.45	0.02	0.82	0.98	0.45	0.18	0.55
BusinessAdministrators	0.40	0.50	1.00	0.20	0.00	0.60	0.80	0.50
Carpenters	0.76	0.05	0.60	0.52	0.40	0.24	0.48	0.95
ChildCare	0.63	0.41	0.42	0.46	0.58	0.37	0.54	0.59
Clergy	0.66	0.46	0.27	0.67	0.73	0.34	0.33	0.54
ComputerConsultants	0.40	0.50	1.00	0.20	0.00	0.60	0.80	0.50
ComputerProgrammers	0.40	0.58	0.50	0.65	0.50	0.60	0.35	0.42
Consultants	0.55	0.55	0.48	0.55	0.52	0.45	0.45	0.45
CorporateExecutiveOff.	0.40	0.50	1.00	0.20	0.00	0.60	0.80	0.50
Counselors	0.57	0.52	0.43	0.50	0.57	0.43	0.50	0.48
Designers	0.55	0.45	0.40	0.43	0.60	0.45	0.58	0.55
Diplomats	0.19	1.00	0.50	0.63	0.50	0.81	0.38	0.00
Doctors	0.50	0.53	0.65	0.43	0.35	0.50	0.57	0.47
Engineers	0.47	0.55	0.50	0.69	0.50	0.53	0.31	0.45
Entrepreneurs	0.53	0.43	0.56	0.56	0.44	0.47	0.44	0.57
EventsCoordinators	1.00	0.19	0.40	0.52	0.60	0.00	0.48	0.81

Continued on next page

Table B.1 – *Continued from previous page*

Job Name	**ENFJ**	**ENFP**	**ENTJ**	**ENTP**	**ESFJ**	**ESTJ**	**ESFP**	**ESTP**
Facilitators	1.00	0.19	0.40	0.52	0.60	0.00	0.48	0.81
FamilyPracticePhysician	0.60	0.50	0.00	0.85	1.00	0.40	0.15	0.50
FashionDesigners	0.50	0.40	0.81	0.00	0.19	0.50	1.00	0.60
FinancialOfficers	0.00	0.81	0.60	0.52	0.40	1.00	0.48	0.19
ForensicResearch	0.58	0.27	0.42	0.76	0.58	0.42	0.24	0.73
ForestRangers	0.60	0.50	0.76	0.05	0.24	0.40	0.95	0.50
HomeEconomics	0.55	0.45	0.02	0.82	0.98	0.45	0.18	0.55
HumanResources	1.00	0.19	0.40	0.52	0.60	0.00	0.48	0.81
InteriorDecorators	0.50	0.40	0.43	0.40	0.57	0.50	0.60	0.60
Journalists	0.19	1.00	0.50	0.63	0.50	0.81	0.38	0.00
Judges	0.27	0.63	0.66	0.48	0.34	0.73	0.52	0.37
Lawyers	0.37	0.59	0.58	0.58	0.42	0.63	0.42	0.41
Managers	0.41	0.54	0.58	0.45	0.42	0.59	0.55	0.46
MarketingPersonnel	0.65	0.30	0.35	0.74	0.65	0.35	0.26	0.70
Mathematicians	0.40	0.50	0.24	1.00	0.76	0.60	0.00	0.50
Mechanics	0.76	0.05	0.60	0.52	0.40	0.24	0.48	0.95
MilitaryLeaders	0.18	0.72	0.68	0.40	0.32	0.82	0.60	0.28
Musicians	0.60	0.56	0.56	0.40	0.44	0.40	0.60	0.44
Nurses	0.55	0.45	0.02	0.82	0.98	0.45	0.18	0.55
OfficeManagers	0.55	0.45	0.02	0.82	0.98	0.45	0.18	0.55
OrganizationBuilders	0.45	0.55	0.98	0.23	0.02	0.55	0.77	0.45
Paralegals	0.50	0.40	0.05	0.80	0.95	0.50	0.20	0.60
Paramedics	0.81	0.00	0.50	0.43	0.50	0.19	0.58	1.00
ParkRangers	0.40	0.50	0.24	1.00	0.76	0.60	0.00	0.50
PCTechnicians	0.81	0.00	0.50	0.43	0.50	0.19	0.58	1.00
Pediatricians	0.60	0.50	0.76	0.05	0.24	0.40	0.95	0.50
Photographers	0.59	0.43	0.43	0.67	0.57	0.41	0.33	0.57
Pilots	0.76	0.05	0.60	0.52	0.40	0.24	0.48	0.95
Police	0.40	0.40	0.55	0.48	0.45	0.60	0.52	0.60
Politicians	0.60	0.60	0.45	0.57	0.55	0.40	0.43	0.40
Professors	0.39	0.64	0.65	0.49	0.35	0.61	0.51	0.36
Psychiatrists	0.60	0.60	0.45	0.57	0.55	0.40	0.43	0.40
Psychologists	0.58	0.58	0.46	0.57	0.54	0.42	0.43	0.42
SalesRepresentatives	0.63	0.37	0.48	0.51	0.52	0.37	0.49	0.63
Scientists	0.40	0.67	0.47	0.73	0.53	0.60	0.27	0.33
Shopkeepers	0.50	0.40	0.05	0.80	0.95	0.50	0.20	0.60
SocialWorkers	0.63	0.46	0.42	0.48	0.58	0.37	0.52	0.54
StrategicPlanners	0.40	0.50	0.24	1.00	0.76	0.60	0.00	0.50
Teachers	0.51	0.60	0.45	0.53	0.55	0.49	0.47	0.40
TechnicalWriters	0.40	0.50	0.24	1.00	0.76	0.60	0.00	0.50
TelevisionReporters	0.19	1.00	0.50	0.63	0.50	0.81	0.38	0.00
Veterinarians	0.60	0.50	0.76	0.05	0.24	0.40	0.95	0.50
Writers	0.48	0.71	0.44	0.56	0.56	0.52	0.44	0.29

Table B.2: The Introverted Types

Job Name	**ISTP**	**INFJ**	**INTJ**	**INFP**	**ISFJ**	**ISFP**	**INTP**	**ISTJ**
Accountants	0.26	0.21	0.23	0.74	0.77	0.37	0.63	0.79
Actors	0.24	0.39	0.43	0.76	0.57	0.27	0.73	0.61
AdministrativeAssistants	0.43	0.43	0.02	0.57	0.98	0.21	0.79	0.57
Administrators	0.38	0.48	0.05	0.62	0.95	0.24	0.76	0.52
AlternativeHealthCarePr.	0.80	1.00	0.61	0.20	0.39	0.50	0.50	0.00
Artists	0.41	0.51	0.71	0.59	0.29	0.74	0.26	0.49
Athletes	0.98	0.77	0.46	0.03	0.54	0.45	0.55	0.23
Attorneys	0.52	0.52	0.61	0.48	0.39	0.40	0.60	0.48
Bookkeepers	0.43	0.43	0.02	0.57	0.98	0.21	0.79	0.57
BusinessAdministrators	0.57	0.48	0.98	0.43	0.02	0.76	0.24	0.52
Carpenters	1.00	0.80	0.51	0.00	0.49	0.40	0.60	0.20
ChildCare	0.50	0.54	0.45	0.50	0.55	0.58	0.42	0.46
Clergy	0.47	0.59	0.34	0.53	0.66	0.42	0.58	0.41
ComputerConsultants	0.57	0.48	0.98	0.43	0.02	0.76	0.24	0.52
ComputerProgrammers	0.45	0.43	0.50	0.55	0.50	0.38	0.62	0.57
Consultants	0.43	0.52	0.54	0.57	0.46	0.52	0.48	0.48
CorporateExecutiveOff.	0.57	0.48	0.98	0.43	0.02	0.76	0.24	0.52
Counselors	0.40	0.50	0.49	0.60	0.51	0.57	0.43	0.50
Designers	0.43	0.43	0.41	0.57	0.59	0.60	0.40	0.57
Diplomats	0.00	0.20	0.61	1.00	0.39	0.50	0.50	0.80
Doctors	0.48	0.51	0.68	0.52	0.32	0.60	0.40	0.49
Engineers	0.51	0.52	0.52	0.50	0.48	0.35	0.65	0.48
Entrepreneurs	0.60	0.56	0.55	0.40	0.45	0.44	0.56	0.44
EventsCoordinators	0.75	0.95	0.51	0.25	0.49	0.60	0.40	0.05
Facilitators	0.75	0.95	0.51	0.25	0.49	0.60	0.40	0.05
FamilyPracticePhysician	0.38	0.48	0.05	0.62	0.95	0.24	0.76	0.52
FashionDesigners	0.48	0.38	0.78	0.52	0.22	0.95	0.05	0.62
FinancialOfficers	0.20	0.00	0.51	0.80	0.49	0.40	0.60	1.00
ForensicResearch	0.79	0.64	0.35	0.21	0.65	0.20	0.80	0.36
ForestRangers	0.38	0.48	0.83	0.62	0.17	1.00	0.00	0.52
HomeEconomics	0.43	0.43	0.02	0.57	0.98	0.21	0.79	0.57
HumanResources	0.75	0.95	0.51	0.25	0.49	0.60	0.40	0.05
InteriorDecorators	0.48	0.38	0.39	0.52	0.61	0.57	0.43	0.63
Journalists	0.00	0.20	0.61	1.00	0.39	0.50	0.50	0.80
Judges	0.39	0.30	0.62	0.61	0.38	0.50	0.50	0.70
Lawyers	0.45	0.41	0.57	0.55	0.43	0.42	0.58	0.59
Managers	0.44	0.39	0.57	0.56	0.43	0.54	0.46	0.61
MarketingPersonnel	0.71	0.66	0.33	0.29	0.67	0.27	0.73	0.34
Mathematicians	0.57	0.48	0.20	0.43	0.80	0.00	1.00	0.52
Mechanics	1.00	0.80	0.51	0.00	0.49	0.40	0.60	0.20
MilitaryLeaders	0.27	0.18	0.65	0.73	0.35	0.57	0.43	0.82
Musicians	0.38	0.54	0.65	0.62	0.35	0.70	0.30	0.46

Continued on next page

Table B.2 – *Continued from previous page*

Job Name	**ISTP**	**INFJ**	**INTJ**	**INFP**	**ISFJ**	**ISFP**	**INTP**	**ISTJ**
Nurses	0.43	0.43	0.02	0.57	0.98	0.21	0.79	0.57
OfficeManagers	0.43	0.43	0.02	0.57	0.98	0.21	0.79	0.57
OrganizationBuilders	0.52	0.52	1.00	0.48	0.00	0.79	0.21	0.48
Paralegals	0.48	0.38	0.00	0.52	1.00	0.19	0.81	0.63
Paramedics	0.95	0.75	0.41	0.05	0.59	0.50	0.50	0.25
ParkRangers	0.57	0.48	0.20	0.43	0.80	0.00	1.00	0.52
PCTechnicians	0.95	0.75	0.41	0.05	0.59	0.50	0.50	0.25
Pediatricians	0.38	0.48	0.83	0.62	0.17	1.00	0.00	0.52
Photographers	0.58	0.61	0.46	0.42	0.54	0.38	0.63	0.39
Pilots	1.00	0.80	0.51	0.00	0.49	0.40	0.60	0.20
Police	0.57	0.38	0.46	0.43	0.54	0.45	0.55	0.63
Politicians	0.38	0.57	0.56	0.62	0.44	0.55	0.45	0.43
Professors	0.39	0.42	0.68	0.61	0.32	0.54	0.46	0.58
Psychiatrists	0.38	0.57	0.56	0.62	0.44	0.55	0.45	0.43
Psychologists	0.39	0.56	0.55	0.61	0.45	0.54	0.46	0.44
SalesRepresentatives	0.60	0.60	0.50	0.40	0.50	0.52	0.48	0.40
Scientists	0.38	0.46	0.52	0.62	0.48	0.34	0.66	0.54
Shopkeepers	0.48	0.38	0.00	0.52	1.00	0.19	0.81	0.63
SocialWorkers	0.46	0.54	0.47	0.54	0.53	0.58	0.42	0.46
StrategicPlanners	0.57	0.48	0.20	0.43	0.80	0.00	1.00	0.52
Teachers	0.35	0.46	0.52	0.65	0.48	0.55	0.45	0.54
TechnicalWriters	0.57	0.48	0.20	0.43	0.80	0.00	1.00	0.52
TelevisionReporters	0.00	0.20	0.61	1.00	0.39	0.50	0.50	0.80
Veterinarians	0.38	0.48	0.83	0.62	0.17	1.00	0.00	0.52
Writers	0.23	0.43	0.54	0.77	0.46	0.56	0.44	0.57

Appendix C

Game scenes

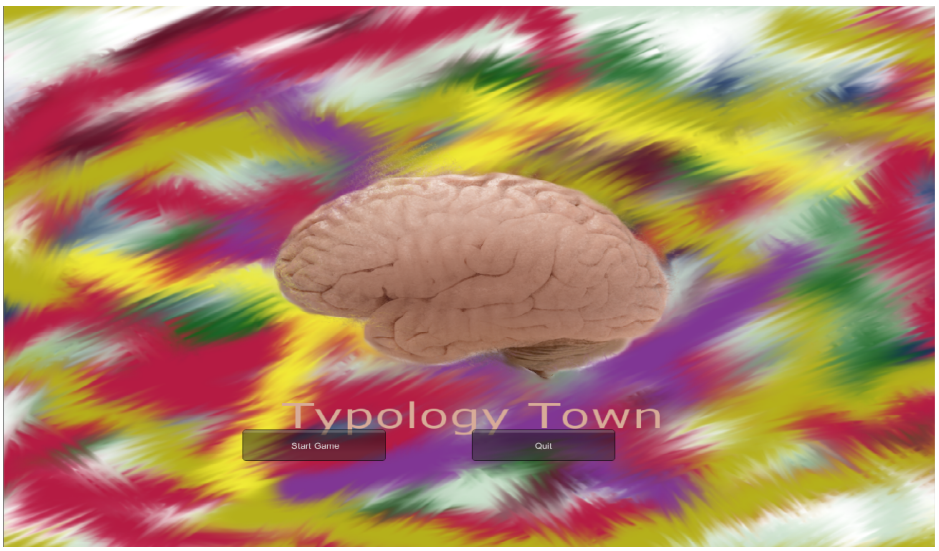


Figure C.1: First scene of the game

Figure C.2: First character personality selection menu

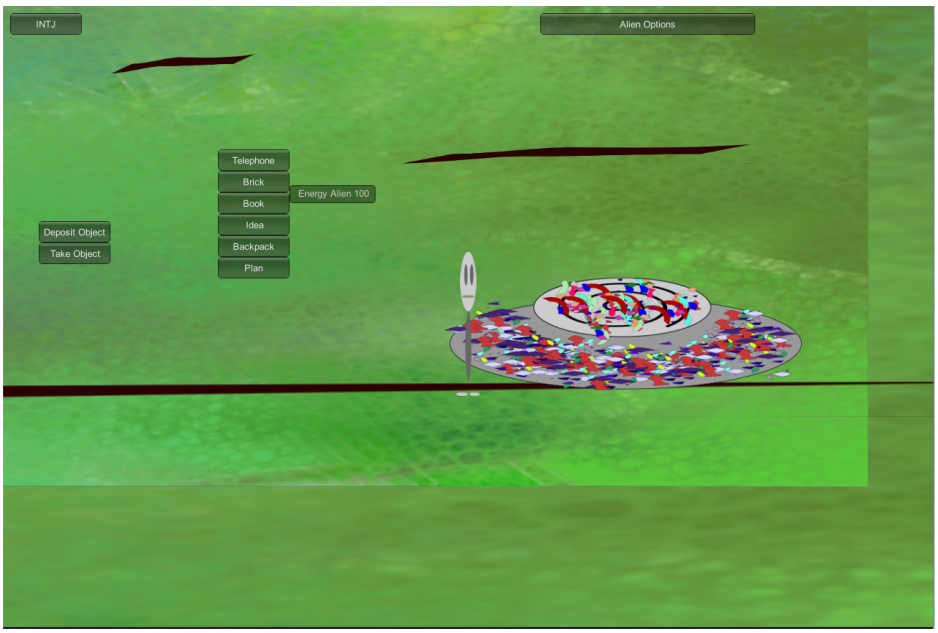


Figure C.3: SpaceShip menu options

Figure C.4: Character menu

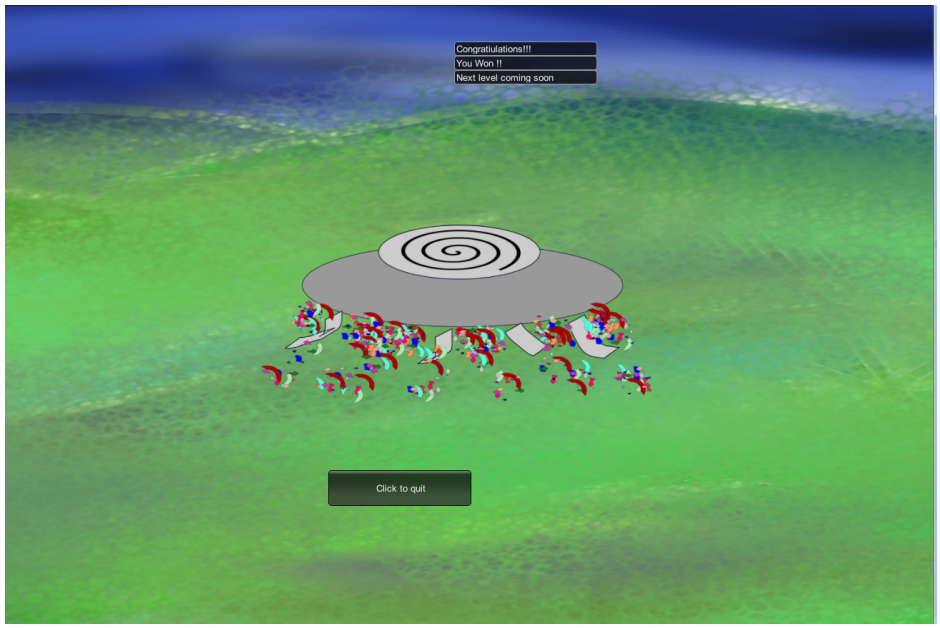


Figure C.5: Game Won

Figure C.6: Game Lost

Bibliography

[1] www.personalitypage.com, 2014.

[2] John Beebe. Evolving the eight-function model. *Psychological Type Review*, 8(1):39–43, 2006.

[3] G J Boyle. Myers-Briggs Type Indicator (MBTI): Some Psychometric Limitations. *Review Literature And Arts Of The Americas*, 30:71–71, 1995.

[4] John H. Bradley and Frederic J. Hebert. The effect of personality type on team performance. *Journal of Management Development*, 16(5):337–353, 1997.

[5] Luiz Fernando Capretz. Personality types in software engineering. *International Journal of Human Computer Studies*, 58:207–214, 2003.

[6] J G Carlson. Recent assessments of the Myers-Briggs Type Indicator. *Journal of personality assessment*, 49:356–365, 1985.

[7] M Carlyn. An assessment of the Myers-Briggs Type indicator. *Journal of personality assessment*, 41:461–473, 1977.

[8] Philip R. Cohen and Hector J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42(1990):213–261, 1990.

[9] Chris Crawford. *The Art of Computer Game Design*. Washington State University, 1997.

[10] M Dastani. 2APL: a practical agent programming language. *Autonomous agents and multi-agent systems*, 2008.

[11] Mehdi Dastani and John-jules Ch Meyer. A Practical Agent Programming Language. pages 107–123, 2008.

[12] Mehdi Dastani and Bas Testerink. From Multi-Agent Programming to Object Oriented Design Patterns. In *International workshop on Engineering Multi-Agent Systems (EMAS 2014)*, 2014.

[13] Sebastian Deterding and Dan Dixon. From Game Design Elements to Gamefulness Defining "Gamification". In *MindTrek conference*, pages 9–15, 2011.

[14] J M Digman. Personality Structure: Emergence of the Five-Factor Model. *Annual Review of Psychology*, 41:417–440, 1990.

[15] C. Elverdam and E. Aarseth. Game Classification and Game Design: Construction Through Critical Analysis. *Games and Culture*, 2:3–22, 2007.

[16] M Eskelinen. Towards computer game studies. *Digital Creativity*, 12(3):175–183, 2001.

[17] Rm Felder and R Brent. Understanding student differences. *Journal of engineering education*, 94(1):57–72, 2005.

[18] Charles W Ginn and Donald L Sexton. A COMPARISON OF THE PERSONALITY TYPE DIMENSIONS OF THE 1987 FOUNDERSKEOs WITH THOSE OF SLOWER-GROWTH FIRM ! 3. *Journal of Business Venturing*, 5:313–326, 1990.

[19] A Hammer. *Introduction to Type and Careers*. CPP Inc., 1993.

[20] Koen V. Hindriks, Frank S. de Boer, Wiebe Van Der Hoek, and John-Jules Ch. Meyer. Agent Programming in 3APL. *Autonomous Agents and Multi-Agent Systems*, 2:357–401, 1999.

[21] Ross Hudson and Don Riso. *Understanding the enneagram : the practical guide to personality types*. Houghton Mifflin Harcourt, 2000.

[22] C.G. Jung. *"Psychological Types". Collected Works of C.G. Jung*, volume 6. 01. mar 20 edition, 1921.

[23] John E. Laird. Research in human-level AI using computer games. *Communications of the ACM*, 45(1), 2002.

[24] Sohye Lim and Byron Reeves. Computer agents versus avatars: Responses to interactive game characters controlled by a computer or other player. *International Journal of Human Computer Studies*, 68(1-2):57–68, 2010.

[25] William McDOUGALL. of the Words Character and Personality. *Journal of Personality*, 1(1):3–16, 1932.

[26] John-Jules Meyer, Jan Broersen, and Andreas Herzig. BDI logics. *Handbook of Logics for Knowledge and Belief*, pages 1–32, 2013.

[27] Michael E. Bratman. *Intention, Plans, and Practical Reason*. Harvard University Press, 1999.

[28] Marc Ponsen, Pieter Spronck, Héctor Muñoz Avila, and David W. Aha. Knowledge acquisition for adaptive game AI. *Science of Computer Programming*, 67:59–75, 2007.

[29] A.S. Rao and M.P. Georgeff. Modeling rational agents within a BDI-architecture. *Readings in agents*, pages 317–328, 1997.

[30] Judy Robertson and Cathrin Howells. Computer game design: Opportunities for successful learning. *Computers and Education*, 50:559–578, 2008.

[31] Michael T. Robinson. www.careerplanner.com, 2015.

[32] Jonathan Schaeffer and H. Jaap Van den Herik. Games, computers, and artificial intelligence. *Artificial Intelligence*, 134:1–7, 2002.

[33] Benjamin Schneider, D. Brent Smith, Sylvester Taylor, and John Fleenor. Personality and organizations: A test of the homogeneity of personality hypothesis. *Journal of Applied Psychology*, 83(3):462–470, 1998.

[34] Alan Thorn. Learn Unity for 2D Game Development. 2013.

[35] Alan Thorn. *Pro Unity Game Developement with C#*. Apress, London, 2014.

[36] Jerome P Wagner and Ronald E Walker. Reliability and Validity Study of a Sufi Personality Typology : the Enneagram. *Journal of Clinical Psychology*, 39(5):712–717, 1983.

[37] Douglass J. Wilde. *Teamology: The construction and organization of effective teams*. 2009.

[38] Douglass J. Wilde. *Jung's personality theory quantified*. 2011.

[39] Michael Wooldridge. *An introduction to multi-agent systems*, volume 365. John Wiley & Sons, Ltd, 2009 edition, 2009.