DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Development of an Emotion Engine for Games

Ibrahim Asif Mirza

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Development of an Emotion Engine for Games

Entwicklung einer Emotion-Engine für Games

Author:	Ibrahim Asif Mirza
Supervisor:	Prof. Gudrun Klinker, Ph.D.
Advisor:	Daniel Dyrda, M.Sc.
Submission Date:	15 April 2021

I confirm that this master's thesis in informatics is my own work and I have documented all sources and material used.

Munich, Ibrahim Asif Mirza

Acknowledgments

I would like to start by thanking my family, especially my parents for their love and support. I would also like thank my advisor Daniel Dyrda for his guidance and support during the entirety of this thesis.

Abstract

Emotion engines provide a unique paradigm to game development. Using computational models of emotion, rooted in psychology, they provide an emotion-oriented approach to game development. Current emotion engines, however, are limited to adapting only narrative elements of a video game. These mostly relate to the story and, to some extent, the dialogues of the characters. Most current video games, though heavily reliant on stories, are becoming more and more visually photorealistic. Each passing year provides new advancements in the tools and techniques used for creating those visuals. Most emotion engines do not take into account these visual aspects of video games. This thesis will set out to conceptualize, design and implement a framework that will attempt to adapt visual as well as narrative elements of a video game. After designing the framework, we will attempt to develop a prototypical game based around the framework. Finally, we shall compare our framework to current emotion engines and frameworks, discuss its shortcomings and shed light on potential improvements and suggestions for further development.

Contents

1 Introduction

Can machines think?

Alan Turing

Alan Turing proposed his famous question "Can Machines Think?" in a paper [101] following a thought experiment. He asked the reader to imagine three rooms and three people: a man, a woman and an interrogator. Each one would be placed separately in the three rooms. The interrogator would then ask questions, using only a single means of communication, and decide which person is in which room. Turing then proposed to replace either the man or the woman with an "intelligent" machine and then repeat the experiment. Would the interrogator then be able to determine whether he is talking to a human or a machine?

In the same paper, he also described various opposing opinions he received. One such opinion dealt with a machine's limited abilities: there are some things that a machine simply can not do. He referenced one put forward by Lady Lovelace [98], that a computer can made be to do one thing and one thing only, but it can not be made to be kind, friendly, have as much diversity of behavior as a man etc.

However, if a computer can be made to do only one thing, and assuming it can do that efficiently, then surely if the computer would be programmed to be "kind" or "friendly", then theoretically it should be possible. Since being kind and friendly are basic emotional states [40, 41] we can then shift Turing's question to "can machines feel?" or "can machines display basic emotions?"

The emotional circuitry of humans is quite profound. According to LeDoux [48] and Misslin [60], our fear and defense systems are innate to our species, acting as the basis to our survival. Misslin [60] further describes the "fight-or-flight" feeling our brains bring up whenever there are external threatening stimuli. Depending on whether there is a fight-or-flight situation or a situation where someone can feel pleasure, this affects our emotional well-being [104].

Since emotions are powerful to us humans, we can propose our own question: Can machines be programmed to exhibit certain emotional capacities in such a way that they end up influencing our own emotional states? In other words: "can we program *machines* to make *us* feel certain emotions?"

Art has been a profound factor in influencing our emotional states. Books, paintings, cinema etc. have been used as profound mediums to tell emotionally driven stories. One such medium came from computers themselves in the form of video games. Utilizing the "theory of fun" [45] and art in its most basic form, computer games have been influential in redefining the status quo and have now gained considerable influence as art forms.

Currently, adding emotional elements to video games is limited to the creative capabilities of the game designers and writers. As an example, games that are heavily story-driven, such as Fallout 4 [25], there would be moments in the story that would affect the mood of the game. This would come in the form of "good" or "bad" choices that the player can make. A "good" choice can result in the player character gaining some new allies or new powers that prove useful for them. Or they can make a "bad" choice, which might result in the player character losing allies and powers, basically "impaling" them.

Depending on the choice the player makes, game designers would then set rules for NPCs (Non-Playable Characters) would act in what way and how would the environment look like. However, it still limited to the skills of the designer, and each designer has their own way of setting rules; there is no fixed way to set these rules.

I propose a framework to allow game designers an emotion-oriented approach to setting the rules of any event from the main story. This will allow game designers to set the emotional rules only once and allow game programmers to implement those rules. The framework, which will be termed as an "Emotion Engine" will then take an incoming external emotional stimuli from an event of the story and based on the personalty each emotional context, will update the emotional state, or the mood, to a new emotional state.

2 Related Work

Human behavior flows from three main sources: desire, emotion, and knowledge.

Plato

This chapter focuses on several research done on the development of models of emotion. Majority of each research stems from the fields of neuro-science and cognitive sciences. We shall also discuss research done by computer scientists in development of computational models of emotion. This will also include software engineering aspects of system development, and how they relate to emotion engines. In addition, we shall touch upon the fields of art, psychology and the importance of aesthetics. Finally, we shall discuss research done upon on using these emotion models in the context of video games.

2.1 A Brief Theory of Emotions

As mentioned before, emotions play an integral part in the survival of humans [48, 60, 26]. Humans have always used various forms of art to tell stories, to convey a certain emotion and as a way to make sense of their lives [58, 29]. This made it crucial for scientists to conduct research on emotions. Sure enough, through decades of research, several theories of emotion have been developed. In the following sections, we shall briefly discuss a few.

2.1.1 Biological and Evolutionary Aspects of Emotions

To understand emotions, we have to understand them at two levels [22].

The first level deals with them on the basis of biology, chemistry and physiology. According to MacLean [50], human emotions are a result of the complex interaction between three brain types, not just one: the reptilian brain, the paleo-mammalian brain and the neo-mammalian brain. These three brain types, as suggested, are as a result of various evolutionary processes.

The second level deals with psychological interpretations from the physiological states of emotions. This incorporates the evaluation of certain emotions based on the

current state of the agent and the corresponding surrounding environment. The final affective states are then categorized. Appraisal theories [47, 67] deal with models on this level. They group emotional states into positive or negative sub-states and continue decomposing them according to some discrete characteristics.

According to Shaver et al. [91], there are six basic emotional states: love, joy, anger, sadness, fear and surprise. They also argue that although emotional experiences are very distinct in a variety of cultural situations, these six basic emotions are considered universal throughout a variety of emotion theories [67, 76, 23]. This also implies that these six basic emotions can be "mixed-and-matched" to give rise to new emotions. In fact, Plutchik's Wheel of Emotions [76] (see Figure 2.1) does a good job in visually explaining this concept. Plutchik suggested eight basic polar-opposite emotions and categorizes them based on color and the intensity of colour.

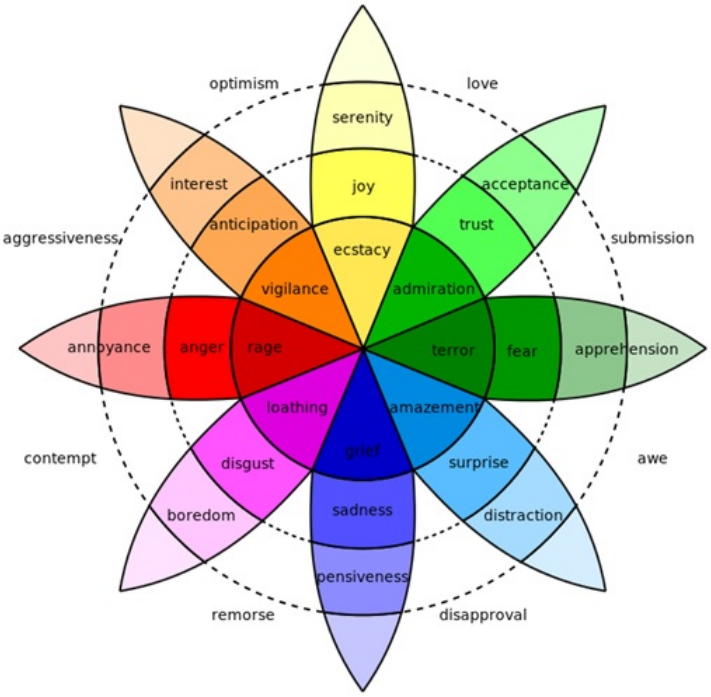

Figure 2.1: Plutchik's Wheel of Emotions machine elf ref[22]

Basic emotions are thought to be universal and primal. They relate to the most basic and instinctual of needs of the body [88]. A great example is "fear", which has been crucial for survival. It works by eliciting defensive responses, such as flight-or-fight, and has been preserved via evolution [66]. Although these emotions are ancient and low-level, there some high-level emotions as well. Those emotions are termed as

aesthetic emotions, and are related to higher needs of our human experiences [71].

Biologically, according to [31], there are neuromodulators associated with each basic emotion. Figure 2.2 shows a model of the the three core affects (stress, dopamine and punishment) which corresponding to fear, joy and sadness.

Figure 2.2: Three-primary-color model of core affects. The three core affects constitute the basic emotions: stress-fear and anger, reward-happiness or joy, punishment-sadness or disgust. [31]

2.1.2 Mood and Personality

To further establish what an emotion is, two related topics, "mood" and "personality" need to be defined. Simply put, they both are utilized in the formation of different emotions [94]. According to Russel [86], emotional events can be thought of states, which influence the person's cognitive behaviours. So, mood can be thought of as the current state of all affective emotional states [54] i.e. the current emotional state of the person. From McCrae et al. [59], the *personality* of a person consists of behavioral traits that the person exhibits in the long-term. Examples of these, according to the OCEAN model [59], are conscientiousness (self-awareness), openness and agreeableness (how cooperative a person is to another).

In other words, mood is the current emotional state of the person, while the personality of the person are traits that determines how the mood of the person will be after an emotional event occurs.

2.1.3 Discrete Theories

The discrete theory of emotion argues that there is a fixed set of basic emotions, rooted in biology. Perhaps the most prominent proponent of discrete theory of emotion is

Ekman [23]. He proposes that there are six emotion families: anger, fear, sadness, enjoyment, disgust and surprise. He further classified them into family groups with similar themes. Example, anxiety and panic are both related via the fear family, mainly because of their underlying systems leading to their expressions.

Ekman goes on to explain some of the characteristics of these families. One of these is the need for a *distinctive universal signal*, which refers mostly to different facial expressions used for different emotion families. Another is an *emotion-specific physiology*, which requires elicitation from a unique component of the brain (or the nervous system). A third one is the need for *universal antecedent events*, the argument here being that emotions that help us with fundamental life tasks should have common grounds, leading to the same type of emotion even when the context is different [23].

He further argues that emotions need to have a *quick onset*. The response to the emotion should be fast enough that the person doesn't even think about it before acting on it. In addition to a quick response time, they should also have a *brief duration*. Their emotional response should not last long so as to not be able to respond to rapidly changing events. He further argues that emotions need automatic appraisal. Due to the quick onset of emotions, the appraisal process, which leads to an emotion, must also be very fast. Since we are generally unaware of the appraisal process it must operate automatically. The remaining characteristics are the *automatic appraisal process of emotions*, the *presence of the same emotion family in other primates*, a *coherence among emotional responses* and the *unbidden occurrence* of emotions [23].

2.1.4 Dimensional Theories

Dimensional theories propose that the emotions should be described on a continuous spectrum instead of a discrete one. This continuous spectrum can take the form of several continuous dimensions.

One such examples of a dimensional model of emotion is the *Circumplex Model of Affect* from Russel [85]. He suggested that different words prescribing emotions can arranged in a circular pattern in two dimensions (see Figure 2.3). Using this circle, emotions can described using three basic properties. The first is *pleasure*, on the horizontal axis with pleasure as the most positive ("pleasant") and misery as the most negative ("unpleasant"). The second is *arousal*, on the vertical axis, with arousal ("activation") having the most cognitive activity while sleepiness ("deactivation") having the lowest.

The third property, however, refers to the consequence or the elicitation of the emotion and not the description of the emotion itself. Thus, this this property can be seen as a combination of the pleasure and arousal component, allowing it to classify all other emotions [85].

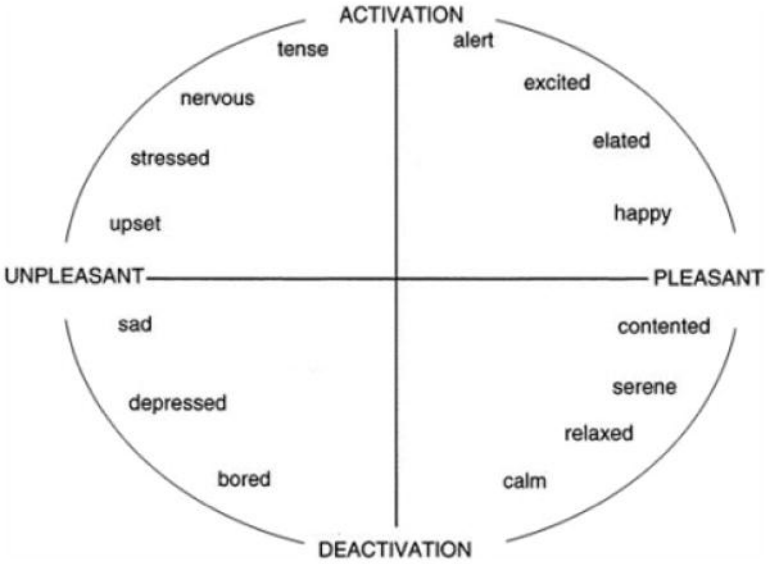

Figure 2.3: A graphical representation of the circumplex model [78]

However, there are certain issues with this model. Russel himself later admitted that only two dimensions are not enough to model all human emotional experiences [86]. Instead, he suggested that *core affect*, perception of *affective quality*, *attributed affect* in combination with *information processing* and *behavioral planning* are all responsible for generating any emotional episode.

Core affect, he explains, is the base emotion in its most primitive form. Core affect is responsible for mood and it influences all other psychological processes based on the current core affect. Example, if the core affect is positive, events that are happening around are perceived to be more positive [86].

Whereas *core affect* is highly personal and subjective, *affective quality* stems from external (or environmental) stimuli. Objects in the environment take up an *affective quality* that influences our behavior towards them and reflect the *core affect* back to us [86].

Objects in the environment can also have an influence on the *core affect* itself, by having an *attributed affect*. The object implies a meaning or a consequence of an event, which results in the *core affect* undergoing a change. The change in the emotional response also allows for variance due to cultural or individual factors [86].

Russel suggest combining both discrete and dimensional views on emotions. He argues that emotional responses are not always dependent on biological or social events, but rather on *core affect*, *affective quality* and *attributed affect*. Those, in turn, are not dependent on each other, rather they can occur independently. And based on the state

(or the mood) of the different components, a mental label is attached or assigned to a specific emotion (like happiness) [86].

2.1.5 Appraisal Theories

Appraisal theories, which are relatively recent, argue that emotions are formed as a result of an ongoing cognitive process in which the current mood of the individual is evaluated. Based on the evaluation, an emotion is elicited [61].

Smith and Lazarus Smith and Lazarus [47] proposed the *cognitive motivational-emotive system*. This theory claims that emotions are a result of cognitive activity, though not all activity is relevant for eliciting an emotion. The process of appraisal is key here; appraisal evaluates whether an event has any impact on the well-being of an individual. This in turn elicits an emotional response in the individual [47] (see Figure 2.4).

They go on to split appraisals into primary and secondary appraisals. Primary appraisals are those that directly concern themselves with the individual's well-being. These contain motivational relevance, whether the event is related to the individual's goals, and motivational congruence, whether the event or encounter brings the individual close to their goal or not [47].

Secondary appraisals concern themselves more with outside factors. *Accountability* determines who gets to be blamed or credited for an event, thereby providing direction for the emotional response and respective coping action. *Problem-focused coping potential* evaluates all possible actions an individual can take to get themselves closer to their goals. *Emotion-focused coping potential* takes the current emotional state (mood) and evaluates all possible gains by adjusting it accordingly to the environment. This leads to *future expectancy*, where emotional changes are evaluated and decided whether they would make the situation desirable for the individual relative to their goals or not [47].

Personality plays a key role in influencing the emotional response of an individual. Since primary appraisals concern themselves with the individual's well-being, here personality concerns with the individual's moral values, goals, commitments etc. With secondary appraisals, beliefs and expectations of an individual's personality play a key role. They also influence the individual's confidence abilities in determine the next step based on the emotional response they receive. This also factors in coping potentials and future expectancy [47].

Both primary and secondary appraisals are responsible for determining whether a situation is relevant to an individual's goals and motivations or not. This also involves whether the individual should adapt to or act upon it. This process results in an emotional state, which prepares the individual to react and cope with the current situation. Since personality play a big role in appraisals, the coping action results

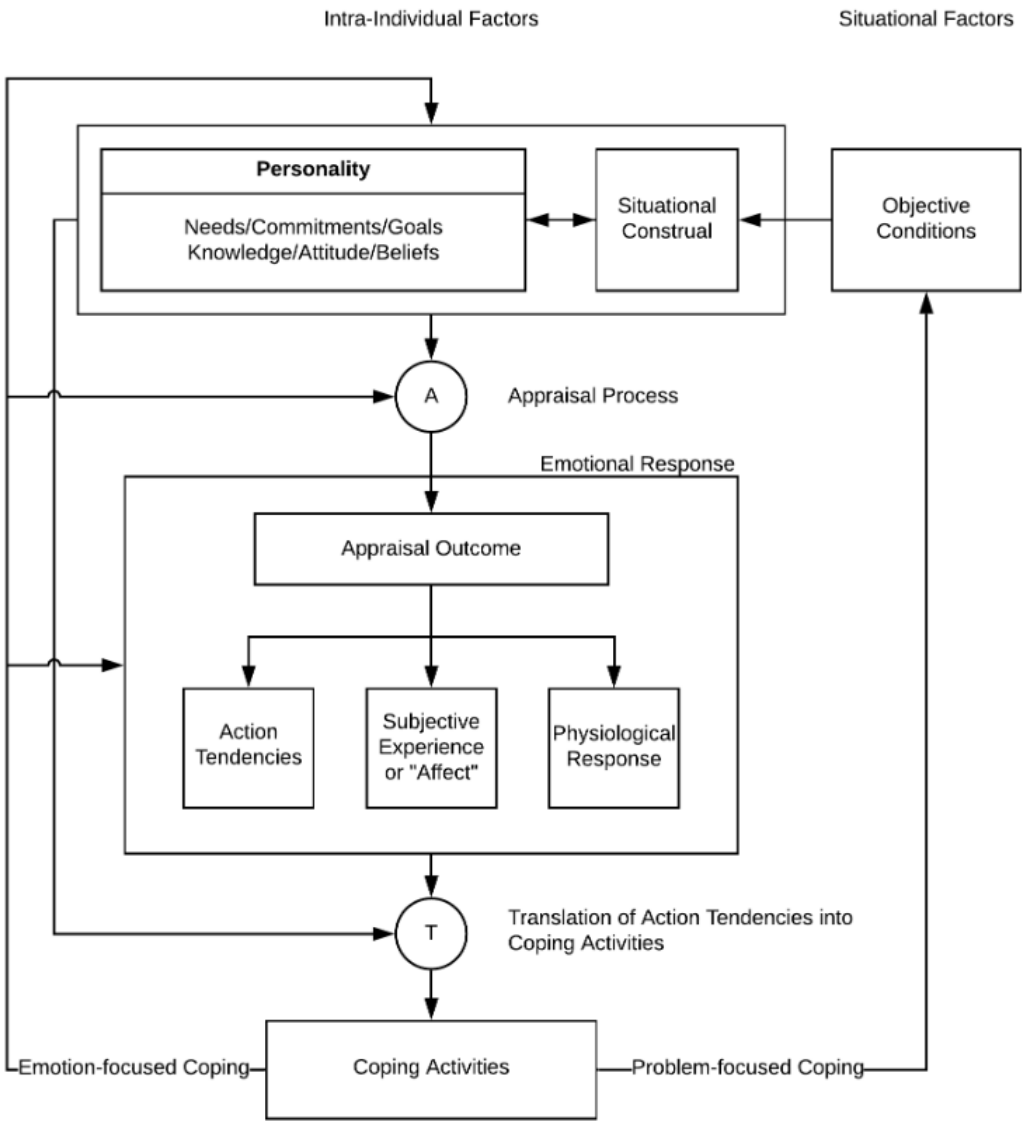

Figure 2.4: Model of the cognitive-motivational-emotive system [46, 47]

not only from the emotional response, but is also influenced by the personality. Not only that, but cultural background of an individual is also factored in to determine whether an action is deemed acceptable or not. Thus, for a full emotional response, both biological and socio-cognitive parts of the individual are required.

OCC Model Another appraisal theory was proposed by Ortony, Clore and Collins, commonly known as the *OCC Model* [67]. This theory was developed keeping computational concepts in mind, and so became a popular theory amongst computer scientists (see Figure 2.5).

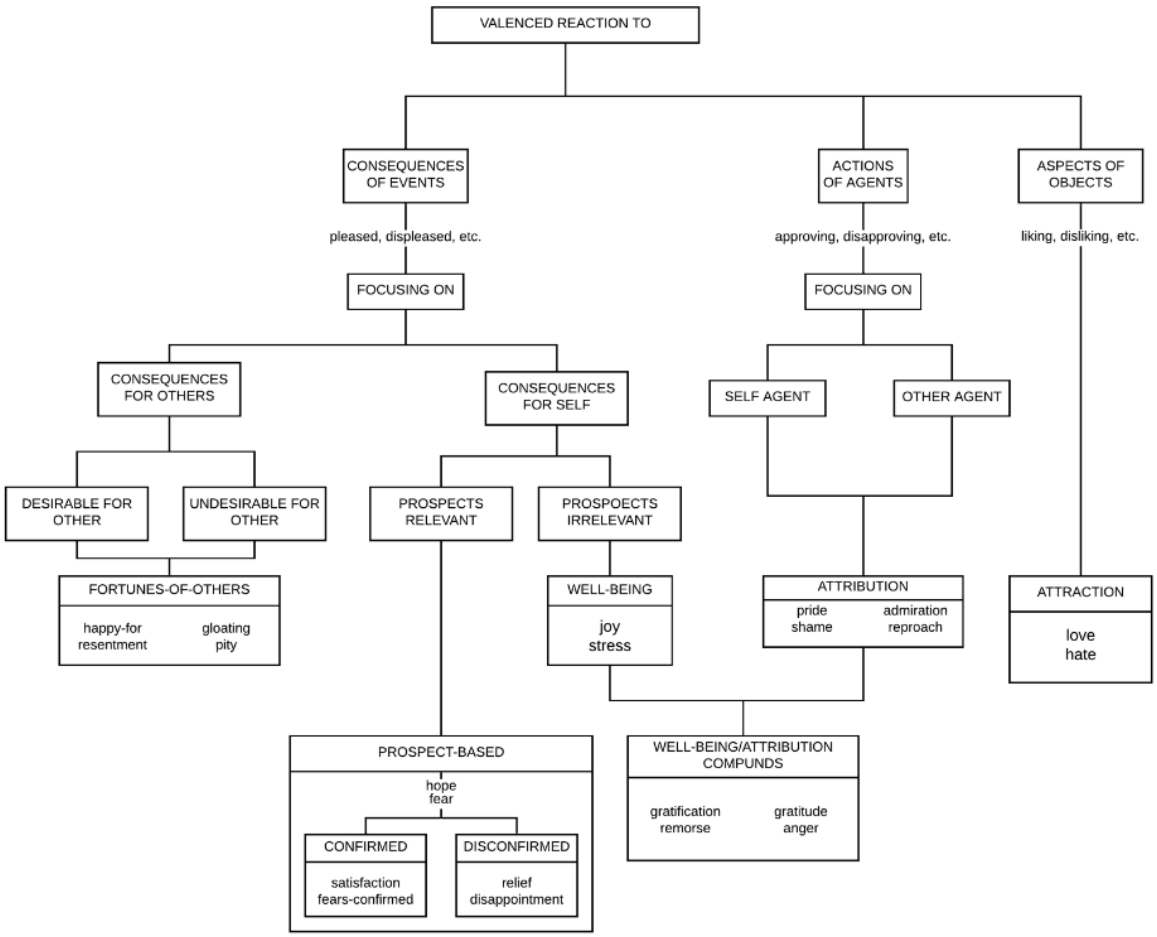

Figure 2.5: OCC Model [46, 67]

Like Ekman's emotion families [23], the *OCC Model* describes three classes of emotion. The model itself consist of 22 individual emotions, but are controlled by three central variables. *Desirability* determines whether the situation supports or works against our goals. *Praiseworthiness* concerns itself with how actions of agents involved and whether they are according to the values held by the subject or not. Finally, *appealingness* deals

with the attribute a person has towards a object [67].

The intensity of emotions, apart from these variables, is affected by global and local factors or variables. Global variables affect all emotions. *Sense of reality* is related to whether the situation appears real or not to the subject. That perception is subjective in itself as well, since, as an example, dreams are not tangible but would appear very real to the subject. *Proximity* is how psychologically close we are to a certain emotional response. Memories are great examples of these, as some memories are so strong that they can mentally transport us back to that time and place. *Unexpectedness* deals with the element of surprised and finally *arousal* deals with how emotionally charged the subject is prior to a situation [67].

On the other hand, local variables affect only those that are in the same emotions group. Likelihood, effort and realization are from the reactions-to-events class. Desirability-for-other, liking and deserving from the fortune-of-others class. Praiseworthiness, strength of cognitive unit and expectation deviation are from the reaction-to-agents class. Finally, in the reaction-to-objects class, there is the attraction and appealingness variable [67].

However, there is still the possibility that an emotion would arise even without these variables. As long as one of the central variables (desirability, praiseworthiness or appealingness) from the corresponding emotion class is specified, it is enough to elicit an emotion [67].

The OCC Computational Model As mentioned before, because of its framework-like structure, computer scientists refined the OCC and adapted it into a computational framework [97]. The original OCC models had holes in it underlying structure, making it quite ambiguous. Computer scientists, however, took rearranged the original model and made strong connections between different emotion type specifications and their underlying logical structure. This resulted in a object-oriented styled inheritance diagram (see Figure 2.6). This also leads us to the next section, which deals with different computational models based on emotion theories.

2.2 Computational Models of Emotion

To have a machine simulate an emotion, a computational framework is need. Certainly, there are several such frameworks that were developed based on the theories of emotion described previously. For that, the ambiguities left in those theories need to be filled first, and each and every step be concretely and explicitly defined and described. So far, most computational models have used appraisal theories as their base model. We shall briefly discuss some of them here.

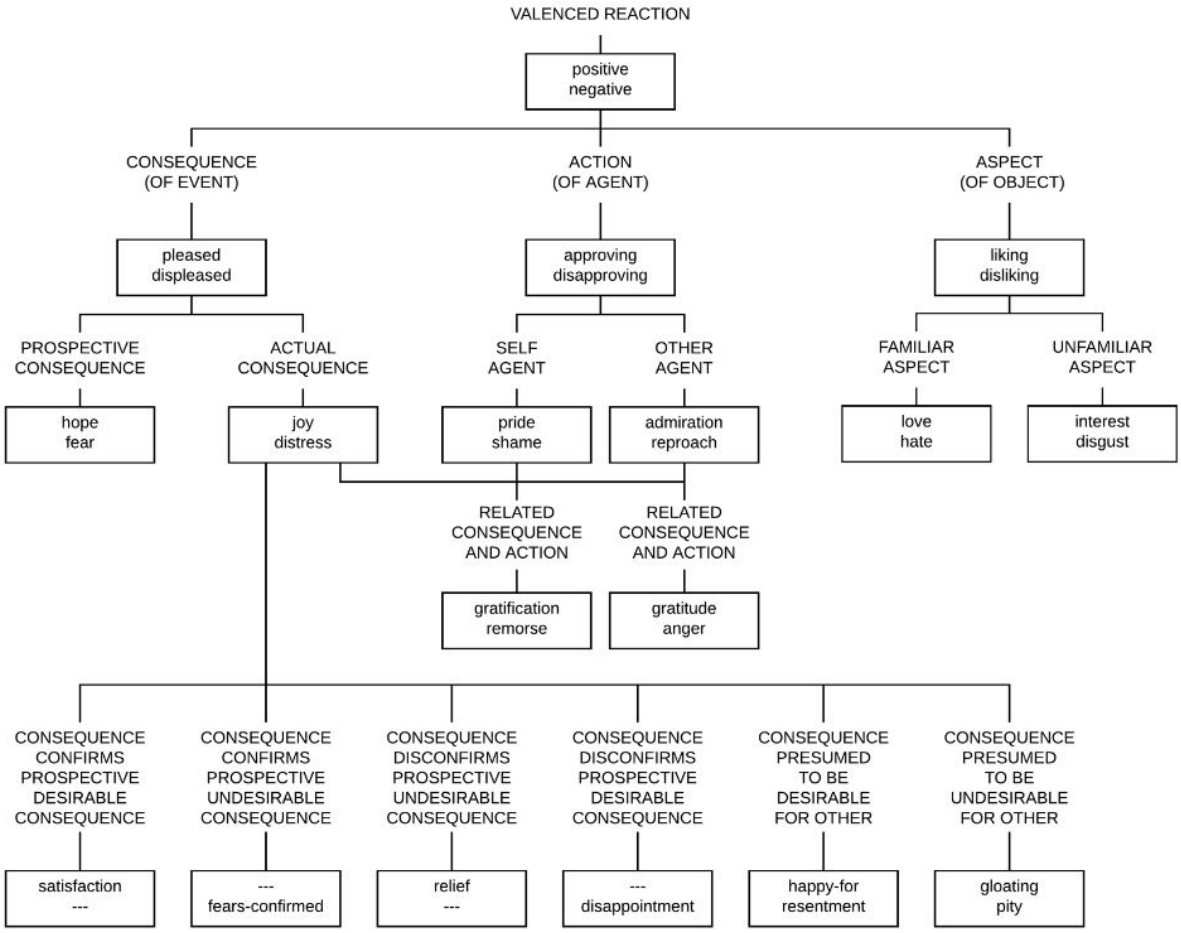

Figure 2.6: Disambiguated, inheritance-based hierarchy of emotions based on the OCC Model [46, 97]

2.2.1 EMA

The EMotion and Adaptations (EMA) model [55], based on the Lazarus and Smith model [47], starts with the appraisal process itself. It begins with the status of the agent-environment relationship, leading to an emotional response, followed by a coping response. The coping response then leads to a change in the agent's relationship with the environment. The system then loops back to the start and begins the appraisal process again [55].

Here, the appraisal process uses the relationship between the agent and the environment as its inputs and outputs. These also include beliefs, desires, intentions, values

and goals of the agent. It works under the assumption that appraisal happens fast and in parallel, as stated in the various theories of emotion. Based on the outcome, the resultant variables include the *perspective* of the agent to the event, the *relevance* and *desirability*. Apart from that, the variables also describe the *likelihood*, *unexpectedness*, *causal attribution*, *controllability* and *changeability* [55].

Thought emotional responses are quick and done in parallel, the mood, however, is set by taking the aggregate of the most intense appraised events, over time [55]. The model itself features different types of coping actions. This is to change the agent's subjective interpretation of the agent-environment relationship. Some them include *attention-related coping*, *belief-related coping*, *desire-related* and *intention-related*. As with appraisals, these coping actions are suggested to be performed in parallel, and are given a sequence to be executed by. This gives it versatility in its processes [55].

2.2.2 MAMID

The MAMID (Methodology for Analysis and Modeling of Individual Differences) is a cognitive affective architecture [38]. It generates emotions and calculates their effect on the behaviors of the agents involved.

The methodology takes into account cognitive and personality traits of the agent. This is based on the OCEAN model [59]. These traits influence the development, intensity, duration and representation of short-term emotions and long-term moods. The system, however, takes into individual differences as well, by treating them as parameters. An introverted agent will likely to experience more intense cognitive activity and fear and will respond more intensely to threats and will add biases to their attention [38].

In its most basic form, the architecture of the MAMID model can be thought of a sequence of three steps: "see, think, do". The environmental cues are taken as input ("see"). These cues are taken to the attention module, where the states of other entities, environment and of the agent itself, are ranged according to their relevance to the agent. This is done by the *situation assessment module* ("think"). It maps the cues to the higher-level *beliefs* held by the agent, which results in the *expectation generation module* projecting all future possible states the situation can find itself in ("do") [38].

2.3 Software Engineering and Emotion Models

The computational models of emotion (CME [68]) described in the previous are just one of many that have been developed. However, the models that were developed usually do not follow software engineering practices. They are mainly driven by theoretical and computational aspects and not from the perspective of a well designed system. A well designed system requires multiple things, including the selection of an appropriate

programming language and the software design techniques utilized to implement the algorithms and the models [75].

Though the research in CMEs has been growing, so far their processes have been quite informal. This is in contrast with software engineering methodologies, which follow a formal and standard methodology. The general structure of the development of a CME has been so far as follows [83, 68, 10]:

Select a theoretical foundation This is where a base theory of emotion is selected around which the computational model will be developed.

Formal interpretation The selected theory is then interpreted using formal languages.

Codification Computer algorithms are then designed and implemented using a programming language based on the interpreted formal language of the theory.

Integration The programmed component is then somehow integrated into a larger software system, as a sub-component.

Validation Finally, the components are then evaluated validated in order to asses whether the emotional stimuli are validated in the system or not.

Software engineering deals with fundamental techniques, methods, practices, and principles that support the development of a software systems [95]. Research in the field of software engineering deals with the design, implementation, testing and maintenance of processes of computer system and most importantly the quality [44]. In essence, a CME is a software system. Because of that, there has been recent efforts to standardize some elements of the development process of computational models of emotion [11].

Taking the four steps stated before, we can adapt the procedure into a standardized formal methodology followed by software engineering principles. According to [68], the following four software engineering phases can be used [9]:

2.3.1 Requirement Analysis

The main objective in this phase is to specify what the emotion model should do. In software engineering this covers discovery, eliciting, organization and analysis of requirements. This also involves defining and describing formal or semi-formal models to specify the software product [51]. It is important to note here that development of computational models of emotion is still conducted within the context of research

projects [54, 53]. In software engineering terms, both the client and the vendor are the researchers themselves. This fact influences the the requirement analysis phase.

Requirements are usually categorized into functional (what the system should do) and non-functional (how to maintain the system). An example of a functional requirement could be that the system should use the OCC model while a non-functional one would be to have the system display a certain notification in less than 50 milliseconds.

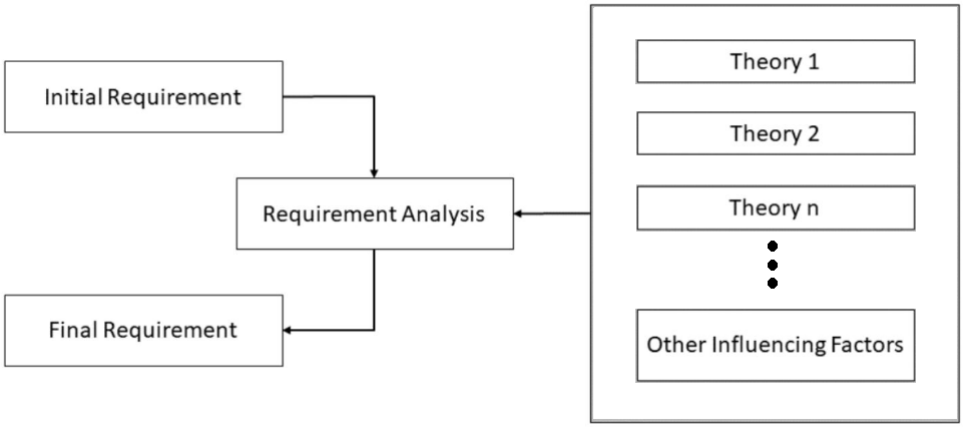

Figure 2.7: Requirements analysis in computational models of emotion [68]

The requirements analysis phase can go through an iterative process, where first the high-level functional requirements are selected. After doing exhaustive analysis, a certain model of emotion can be selected. The selected model, in turn, would be put through the another exhaustive analysis phase in the context of the initial requirements, resulting in an final requirement. This final requirement is an updated version of the initial requirement and contains the decision to either choose another model of emotion or change the requirements altogether (see Figure 2.7).

Requirement analysis in the context of CMEs require focus on the objective of the system. As an example, *domain-Specific CMEs* are conceived within a specific domain (for example video games or education).

A really good example of this is GAMYGDALA, an emotion engine made specifically for utilization in video games [77]. The developers of GAMYGDALA focused on three main requirements. The first one being that Non-Playable Characters (NPCs) should be capable enough to evaluate events on a psychological level. For this, they decided to adopt the OCC Model [67]. The second one being so that users of GAMYGDALA do not need to know much about appraisal theories and the final one being that the CME should be efficient enough to harbor a large number of agents. Developers can then assign objectives to these non-playable characters, which they will then evaluate

upcoming events based on their level of fulfillment of their goals. This will result in the appropriate emotional responses.

Another game related one is ERiSA, which provides a CME for both personality and social skills to non-playable characters in the video game The Elder Scrolls V: Skyrim [17, 16]. For this, the developers opted for five requirements. The first one being that the CME should have a personality component, so the OCEAN personality model was used [59, 18]. Secondly, the CME should allow non-player characters to generate emotions so Ekman's six basic emotion classification was used [23, 24]. Third, the same non-player characters should have their emotion states reset back to their original states via an emotion decay function. Fourth, social relationships should also be taken into account [17]. And finally, there should be some dynamic interaction of the CME with the current state of the game and its rules. The final proposed CME is used a framework for developing social game agents to observe the emotions of user.

The Conscious-Emotional-Learning Tutoring System (CELTS) is another great example of a Domain-specific CME in the context of education. CELTS is capable of building emotional profiles of students to better understand and interact with them. The requirements of CELTS are as follows: First it should keep track of emotional episodes, based on the Ledoux principles [49]. Secondly, it should learn and adapt from the emotion interaction between the tutor and the students. This requires an implementation of emotional learning [6]. Third, the CME must be quick enough to interpret stimuli in a relatively fast pace. Finally, the CME must preform precise stimuli evaluations [13] and make judgements of external stimuli.

In conclusion, these CMEs require an exceptional and deep knowledge on the emotion of theories described in the previous sections. This also means that CMEs can be developed based on any one those theories [103, 5, 36]. They should, however, implement only key mechanisms from those theories (example, mood and personality of agents for dimensional theories) [56] (see Figure 2.8).

2.3.2 Design

This phase is where the architecture of the CME is designed. This involves data flow and computational techniques pertinent to the selected model of emotion. The design aspects, highly influenced by software engineering principles, are based on two key concepts.

Architectural Design

The *architectural design* of the model is a high-level abstraction of the system and is pertinent to translating functional and non-functional requirements into low-level

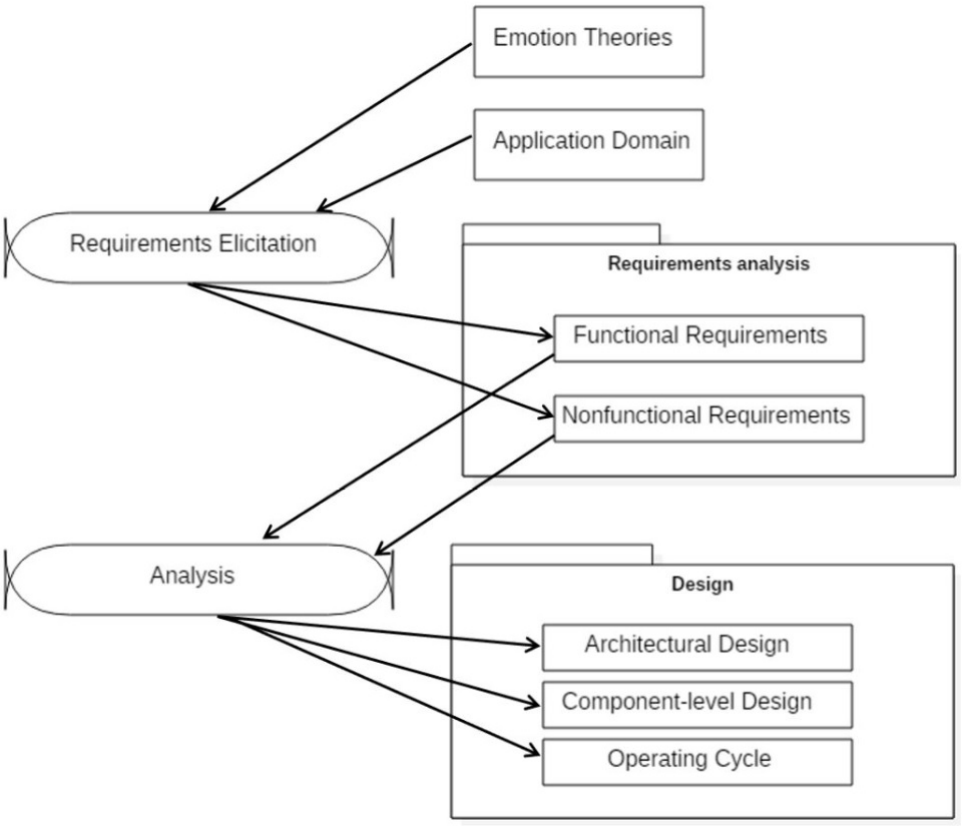

Figure 2.8: Requirements elicitation for the development of a CME, based on [12, 68]

components and data structures that the emotion model will use. The architecture gives an overview of the components and how they are integrated to form a cohesive whole. In other words, they provide a way to structure the design of the model.

Obviously, there is a large repertoire of conventional software architecture designs that we can use. Some examples are data-centric, data-flow, call-and-return, object-oriented, layered, etc.[79]. This gives us a huge advantage as we can use software architecture design tools like the Unified Modeling Language (UML). UML allows us to specify, visualize, build and document the components of a software system [84]. Best of all, it is widely used in modeling software system.

Various researchers have incorporated different architectural designs to facilitate their development process for their CMEs. Deep Emotion [36] is a great example of this, and it contains three main layers. The first one corresponds to the appraisal component and is related to the internal and external evaluation of that appraisal. The second layer is responsible for adjusting the results from the first layer to the surround

environment the agent is in. The third and the last layer utilizes reinforcement and sequential learning for the agent (see Figure 2.9).

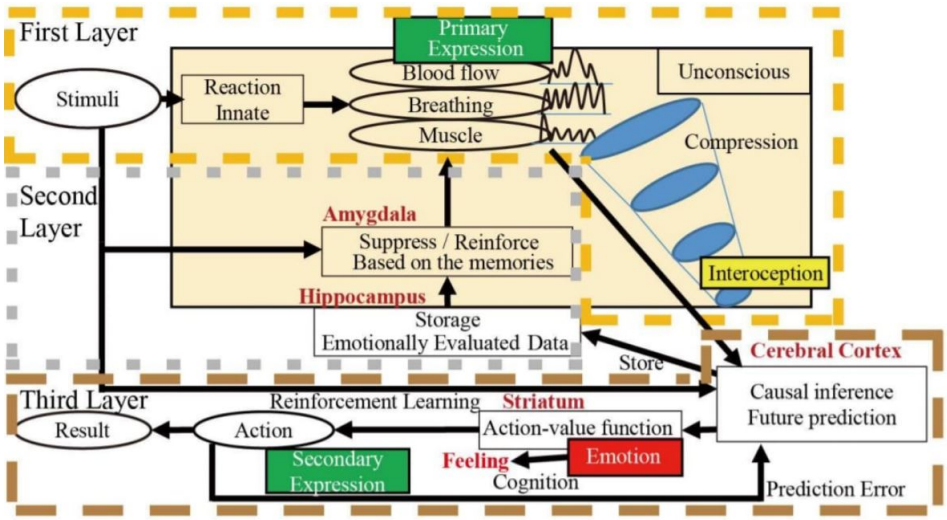

Figure 2.9: Architectural design of DeepEmotion [36, 68]

It is important to note that the architecture in Figure 2.9, even though has a great level of detail, is quite complex. It did however provide guidelines to redesign and refactor this initial architecture and ultimately led to Figure 2.10. Clearly Figure 2.10 is a version that that makes implementation possible.

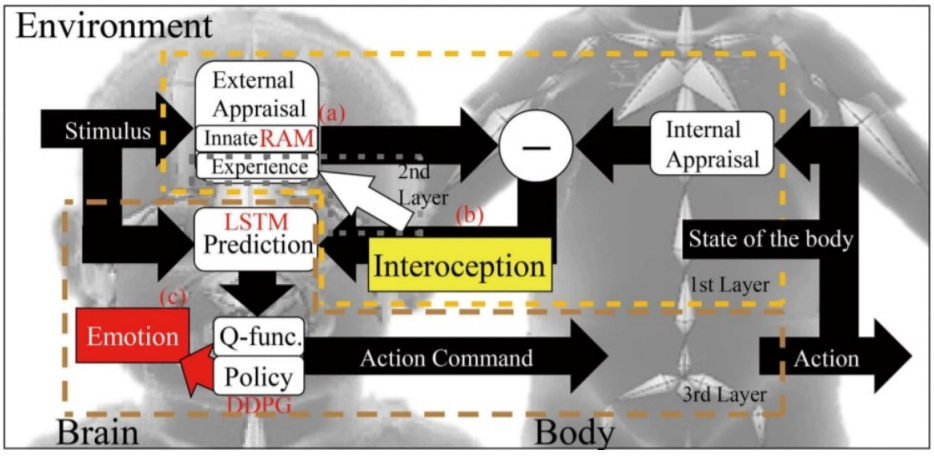

Figure 2.10: (Final) Architectural design of DeepEmotion [36, 68]

Another examples of a CME that followed software architecture principles is FLAME

[63]. One key feature in FLAME is that it uses fuzzy logic to interpret emotions based on their intensity. Figure 2.11 shows the architecture of the agent in FLAME. There are two levels in its design. The first level has three components, mainly the *learning*, *emotional* and the *decision-making* component. The second level deals with individual sub-components. Figure 2.12 shows details of the Emotion Component in FLAME

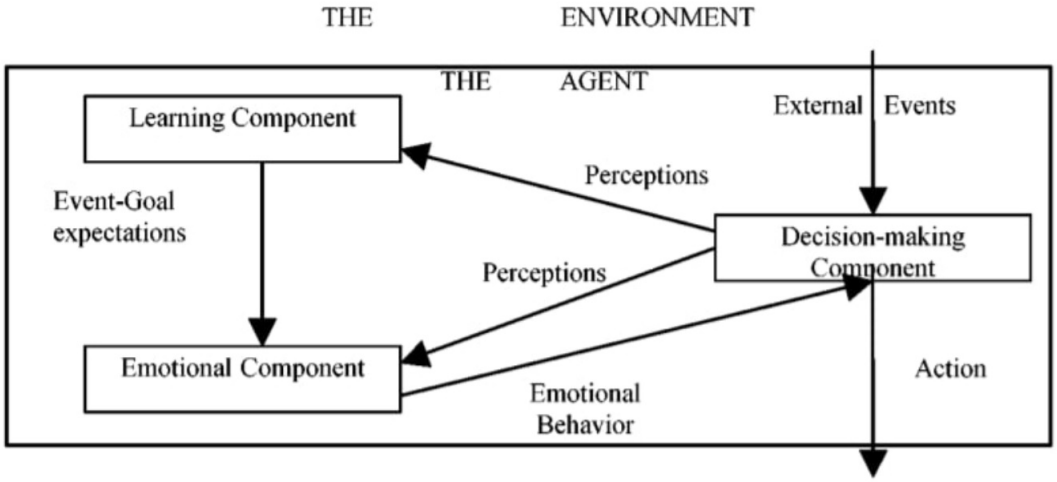

Figure 2.11: General design of the agent architecture in FLAME [63, 68]

As we have seen from the the architecture of these CMEs, their purpose is to give a visual overview of their components, their functionality and the relationships between them. This provides good arguments as to why we should use a component-based approach while stepping towards designing the architecture and its future implementations and improvements.

Component Design

The *component design of the model*, which is a low-level abstraction of the system and is pertinent to refining and and implementing the architectural design of the model in terms of modules, sub-modules and functions, which when combined together, builds up the final algorithmic representation of the CME.

Whereas an architecture works as a whole, a component works independently and only provides an interface for input and output. Since these are still abstractions, meaning no source code is needed, it provides a huge advantage in representing them visually, in terms of software re-usability. Components generally have an interface to *provide* access to their internally stored data and also another interface to *request* access to other components.

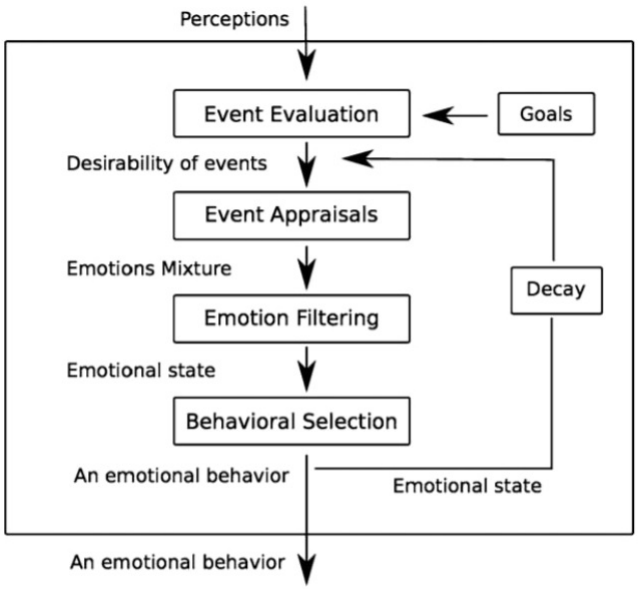

Figure 2.12: Details of the Emotion Component in FLAME [63, 68]

As mentioned previously, FLAME uses a component based architecture to organize and structure its functionality. And each component provides an interface to receive some input and give some output, usually from one component to another (see Figure 2.12). This provides a good argument to use components when it comes to designing CMEs. Based on this, various frameworks have been proposed to follow a component-based approach to designing CMEs. Examples are the Infra (Integrative framework) [82, 14] and EEGS (Ethical Emotion Generation System) [64, 65]. Both Infra and EEGS frameworks follow a component-based approach to developing CMEs and usually contain a list of five to six different module or components. These components or modules usually relate to the interaction between modules, the appraisal process, filtering emotions, affective generation etc.

2.3.3 Implementation

After the architecture has been designed, software tools and programming languages are chosen to implement the design and translate it into code. Any programming paradigm can be used, though most pieces of software are usually developed using an object-oriented paradigm. In any case, software engineering provides several guiding principles that we can following during our coding processes [68].

Software Tools It is important to choose an appropriate integrated development environments (IDE) for the programming language of choice. These IDEs provides features that greatly aid coding processes, such as code completion, enforced structuring of files and folders and even robust debugging tools. IDEs can also be extended to integrate other software tools, so that coding and debugging can be streamlines. Apart from that, software versioning tools, like Git, can be used as well to have a trail of different versions of the software.

Best Practices It is not enough to translate the architecture into code, but also to enforce some rules related to coding itself. Examples of these are documentation of source code and the structure of the code itself.

Standards pertaining to Code It is also important to define some coding standard that will be used consistently and homogeneously throughout the coding process. This will enforce all participants to follow those defined standard, allowing them to understand code in a consistent manner. This can make maintaining code quite easier.

Despite these software engineering principles, there is no evidence that these practices are actually being used in CMEs. There are many CMEs developed that reported some aspect of software engineering [68] (like specifying a programming language) [63, 103, 8], they do not seem to explicitly follow software engineering principles. We can instead refer to the world of open-source to have an "under-the-hood" perspective on their code.

Derek [89] and CakeChat [39] are two open-source CMEs. Derek uses extensive documentation and standard coding practices enforced within the project to have homogenity across objects and variables [89]. The code itself is structured logically. CakeChat utilizes software engineering practices to enforce flexibility and a dynamic environment. Implemented in Python, it allows to tweak the model's responses by a conditional variable. Like Derek, CakeChat is documented, has exception handling and utilizes modularity [39].

2.3.4 Testing

The testing phase occurs after the system has been implemented. It typically involves either running test cases using mock data and validating their results according to pre-defined requirements. Or find situations or conditions where the behavior of the software deviates from what is required. In either case, this is an essential phase, as it allows the designers to ensure the quality of the designed software product.

When it comes to CMEs, it is somewhat difficult to test them. The central object that is being tested is how realistic the dynamics of emotions of an agent are in the system. Usually, researchers would test CMEs by running simulations or test case scenarios and verify whether the system meets the requirements or not. This approach usually involves *defining the test case*, *defining the input/output data*, *running the program against the test case* and *checking the results* [68].

An example of a CME that does testing is PEACTIDM (Perceive, Encode, Attend, Comprehend, Tasking, Intend, Decode, and Motor) [53]. The testing is done by preparing some mock data, which involves placing the agent at a specific position, so that to get to its target, it can only take a specific number of steps. Running the simulation will provide a series of data. This resultant data is then compared with the requirements of the CME to cross-check if the requirements are met or not.

EMA (EMotion and Adaptation) [54], as discussed in a section 2.2.1, uses test-case scenarios. The states, actions and the appraisals executed by all agents involved are defined, as well as the set of stimuli that would incite those states. After the test is run, the resultant appraisal frame and the associated emotion label would be then compared with the pre-defined requirements for EMA to cross-verify the model.

2.4 Aesthetic Emotions

Apart from the primal nature of basic emotions, there is also an additional component. This component has been consciously and unconsciously vital to the survival instinct, and that is *interest* or *curiosity*. There has been several research done [72, 73, 74] that deals with discussion of this instinctual emotion for knowledge, curiosity and *"drive to know the unknown"*.

Berlyne [7] argued that the hedonic qualities of various pieces of art are formed from distinct biological systems. One system, which is the primary reward system, generated positive affects whenever the person's arousal potential increased, whereas the secondary system, which is responsible for aversion, generated negative affects whenever the same arousal potential increased. And each one was triggered by the same instinctual emotion: *interest* or *curiosity*.

Berlyne has done extensive research in "interest in response to art". Tan [32] supports that research by claiming that *interest* is quite instinctual to the "aesthetic experience" of arts. To explore this claim, Silvia [93] conducted several experiments to test appraisals and their connected with "interest" in art. The first experience consisted of showing people randomly generated polygons, based on experimental aesthetic research. The study stared out with simple polygons and then went to display complex polygons, all the while assessing the patrons' ability to understand complex art. When asked to

pick the most interest polygon, it was found that the appraisals of ability to understand complex art constituted to the complexity of the most interesting polygon. People felt more comfortable to pick complex art when they were able to understand what complex art is.

According to Kant [43] the notion of aesthetic experience is that of disinterested pleasure. Fear, for example, is not an aesthetic emotion because it activates an instinctual fight-or-flight state within us. Contrast to this, being emotionally moved is an aesthetic emotion which may involve goosebumps, eye-tears or any other physiological symptom also associated with fear [87]. Silvia [92] argues that art may evoke any kind of emotion. He suggests that all emotional responses are aesthetic.

Complex artwork gives rise to complex aesthetic emotions [21]. Experiencing such an artwork may place the viewer into a pensive and introspective state. This will involve triggering of emotional memories and letting people re-experience the past [21, 30, 52].

2.5 Video Games and Affective Modelling

When it comes to video games, there has been debate as to whether they are considered forms of art or not [3, 69]. Regardless, in video games, because of their interactive nature, emotions play a key role in forming the gameplay mechanics by which a player can experience an immersive atmosphere. This takes the emotional investment to a whole another level. According to Crawford, computer games are art forms that "present [their] audience with fantasy experiences that simulate emotion" [19]. This implies that games have are those type of art forms whose immersion is strongly linked to their associated emotion.

According to Chen [15], a games should be well designed to provide players a feeling of pleasure and happiness in their flow zone. This flow is from Csikszentmihaly [20] (see Figure 2.13), in which a person is completely immersed in an activity via focus, involvement and enjoyment (provided, of course, the right balance between difficulty and enjoyment is achieved) [20, 22].

2.5.1 Adaptive Narratives and Player Modeling

Perhaps the most immersive aspect of video games, apart from the gameplay itself, is the story. According to Abbott [2] and Forster [27] , a story is a sequence of events involving fictional or non-fictional characters. Plot, on the other hand is the logical and causal structure connecting the events. To represent this story and its plot, an appropriate representation is chosen. It could be oral, written, theatrical etc. This representation is the narrative of the story [2].

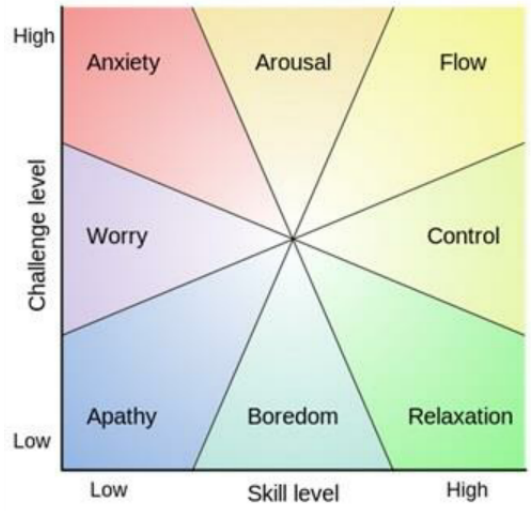

Figure 2.13: Csikszentmihalyi's Flow Model [22, 62]

Since games are interactive in nature, Jenkins [42] puts forward two types of the narrative content: *embedded narratives* and *emergent narratives*. Embedded narrative is pre-generated content by the developers or writers, while emergent narrative is the content generated the fly by the players.

PaSSAGE [100] is an example of a game system that creates a player model based on their playstyle. PaSSAGE then uses that model to adapt the story to the player [100]. The plot of the story is contained inside a library, annotated with their corresponding player types. All this is is annotated by the writer of the story, so related events are grouped together and a common narrative is chosen so as to link all groups. In this case, it is Joseph Campbell's classic myth structure [4]. This is so that different players encountering different events will still have a believable and coherent over-arching story. The developers created a player modeling module that would generate the player model on the fly, taking into account the actions taken by the player inside the game. The module would then calculate the weight of *"interest"* of the next possible story event and show it to the player.

PACE (Player Appraisal Controller Emotions) [33, 34] is another system that generates the player's model based on different playstyles. The system then infers which narrative plot points should be shown to the player, depending on the player's goals.

From this we can gather that game adaptivity starts with defining a player model. This are numeric attributes that describe the playstyle of a player [37], or in other words, the behavior of the player inside the game. According to Sharma et al. [90], there are two approaches to modeling players in video games:

Direct Measurements This approach utilizes psychological and behavioral data of the player by acquiring them via bio-sensors. Peng at al. [70] uses a direct-measurement approach to adapting video animations according to the mood and the emotions of the viewer. Although this was not used in the field of video games, it did adapt character animations, which are considered a key component in video games.

Inferred Measurements This approach utilizes the "Big Five" model or player types and determine if player actions can have an influence on the affective model. However, a disadvantage of this approach is not knowing if player actions can correctly infer player emotions.

2.6 Current Limitations

Current video game system, such as Passage [100] and PACE [33] adapt only the plot of the story when it comes adapting narratives. They do not discuss adapting other aspects of video games, such as environmental atmosphere, lights, cameras, sound, 3d models etc.

Freilão [28] did propose a solution to this issue in their framework. Their proposed framework takes into account not just plot elements, but also color schemes, dialog options and the overall feel of the game. However, it is still limited to the modeling the player via asking the player their emotion state first (using some sort of questionnaire) and then influencing the game's visuals from there.

Their framework also does not account dynamic mood and personality of the player avatar character and NPCs. This is because they first ask the player what their emotional state is, which is then reflected onto NPCs and their dialogues, when in fact it should be dynamic.

ERISA [17, 16] comes close to adding mood and personality to NPCs, but similarly to other frameworks, it is limited to only mood and personality and not the story and other visual elements. We've seen from the discussion of aesthetic emotions that interest, curiosity and the drive for the unknown is important and art, because of its visual nature, is as important an emotion as the six basic ones. However, we have also seen that none of these systems take into account the visual aspect of emotions.

In the next chapter, I propose a framework that will attempt to solve this issue.

3 Emotion Engine Framework

We are not thinking machines that feel,
we are feeling machines that think

António R. Damásio

In the previous section, we talked about emotions, their computational models and how their variations of them are used in different domains (such as video games). We also covered how aesthetic emotions are as important as other emotions and we saw some emotion engines designed for video games. However, we also saw shortcomings in those engines, specifically how limited they were in regards to adapting visual aspects of the game.

The emotion engine framework presented in this chapter will attempt to overcome most of those shortcomings. We would have to come up with an emotion theory (as described in section 2.3.1). We would also have then come up with an appropriate architecture, implementation and ultimately provide a language-agnostic framework.

3.1 Goal of the Framework

The main goal of the framework is to provide a standardized way of adding emotional components based on any emotion theory and software engineering principles. This will help in designing an architectural paradigm for an emotion-oriented approach to game development.

The framework will also provide a way to generate an adaptive narrative not only for the story of the game but also other visual elements of the game. We will develop a prototype based on the framework, which will be discussed in chapter 4. For that, we need to pick a video game genre and narrative style.

3.1.1 Video Game Genre

As described in section 2.5.1, story is quite different from narrative. The same story can be told in different ways and thus there are multiple possible narratives. Examples of some video games are Red Dead Redemption [81] and Uncharted [102]. Both are

action adventure games, but the former has a more open-world structure, revealing the story gradually and through its environment and the latter has a more linear structure, revealing the story like a movie.

For the proposed framework, although it can work in any video game genre, because of time constraints, a text-based adventure game was chosen. Some examples of these type of games are Zork [105] and 80 Days [1]. These games for known for their unique style where the main goal is to read and experience the story and explore. The gameplay is quite simple, reading and interacting with different options that are presented to you, but can also include puzzles, mini-games and battles. These are mostly heavy story driven games and rely on the writing skills of the writer.

3.1.2 Narrative Elements

For the genre discussed in the section 3.1.1, we should also choose narrative elements to adapt. As mentioned before in various previous sections, multiple elements can be adapted (such as camera settings, audio and music settings etc.), but for such a genre, they should be characteristic to that genre. Additionally, the number of elements and their complexity should also be considered.

For the purposes of this thesis and the genre, the chosen narrative elements are:

- Visual Elements
  - Text animations, which consist of animations applied to textual elements
  - Colors, which consist of a dynamic color palette
- Textual Elements
  - Dialogue between characters

3.2 Emotional State as a Computational State

Taking Ekmans' theory [23], we can model an emotional state of an agent as a state chart (see Figure 3.1). Keeping this mind, we can then define emotion, mood and personality as the following:

Emotion Emotion is a unit state in the most basic sense. A unit emotion state represents a unit psychological emotion (like anger, fear, happiness etc.) We can also model emotional states based on different theories of emotion (basic, discrete, continuous etc.). We can also use different values for determining the different magnitude of a specific emotion (see Figure 3.2).

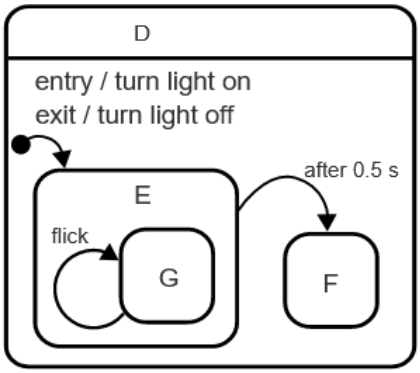

Figure 3.1: Example of a state chart [96]

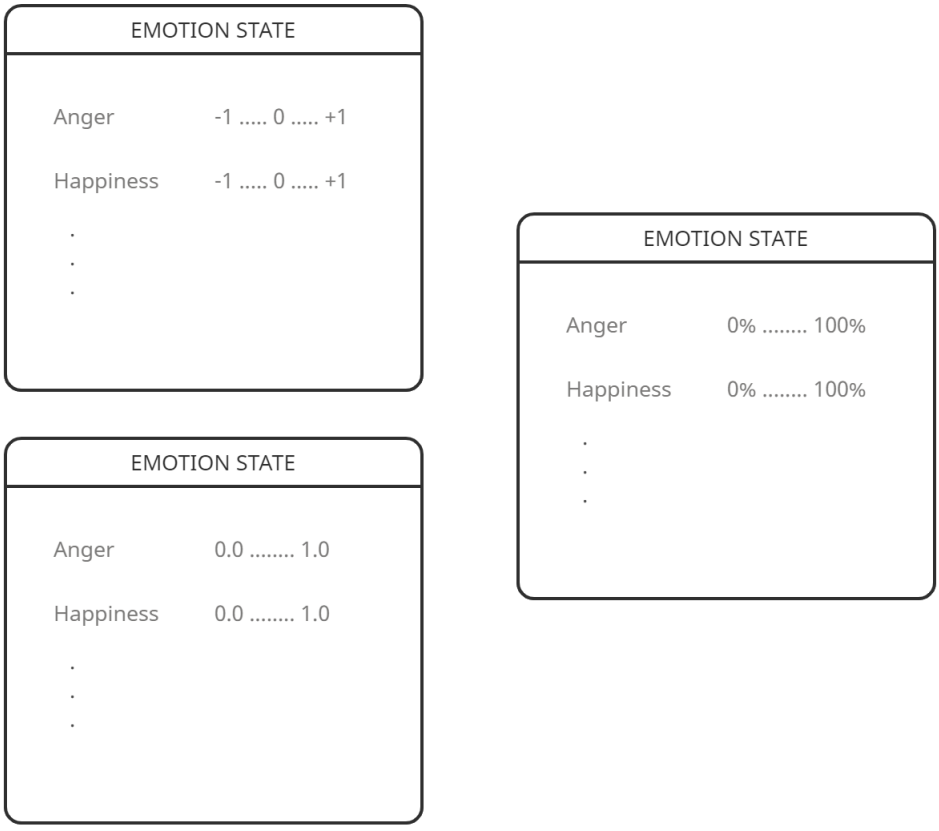

Figure 3.2: Examples of emotional states

Mood Mood represents the current internal state of the agent. In terms of a state-chart, this represents which emotions have been triggered and have been activated (or deactivated) based on some external incoming input stimuli.

Personality Personality can then be described as the mapping or a transitioning function that takes in an external stimuli, which consists of new upcoming emotion states, and activates/deactivates the agent's internal mood state, effectively changing its internal state. The personality mapping function takes this stimuli, do some processing based on the function definition and description, and then updates the internal state of the agent (in this case, it is the mood).

3.3 Personality as a Computational Mapping Function

Thinking about emotions as components that are attached to player objects will provide a good approach to add adaptive features in video games. These emotion components can then be controlled by the emotion engine, where the personality mapping function will update the internal mood state. The internal mood will then gets reflected on to other components of the agent.

As an example of such a component attached to, say, a player avatar, one implementation would be Figure 3.3. Notice that each change made to the other components of the Player avatar is driven mainly by the personality mapping function. The incoming emotional stimuli will trigger the personality mapping function. The function will take that incoming stimuli and activate/deactivate the internal emotional states of the character. The mood is connected to other subcomponents of the character. The updated internal mood, in turn, updates each subcomponent of the character (such as speed of movement, selecting specific lines of dialog etc.)

This also gives an opportunity to expand this state chart to other objects in our video game as well. These objects can be inorganic objects, like buildings, roads, rooms and cars (see Figure 3.5), and even abstract objects, such music, audio systems and even weather systems (see Figure 3.6). Weather systems have been increasingly become more and more popular in recent games, such as Ghost of Tsushima [80] (see figure Figure 3.4). According to [80], the weather changes based on the player's playstyle, and is a really good example of an adaptive narrative system.

Though this might seem like a strange idea to give inorganic objects a "personality" or a "mood", the emotion engine framework will actually work with all objects that have this emotion component. Specifically, their personality will have basic switching methods that will take the single incoming emotional stimuli and then update their internal properties (like colors, positions etc.).

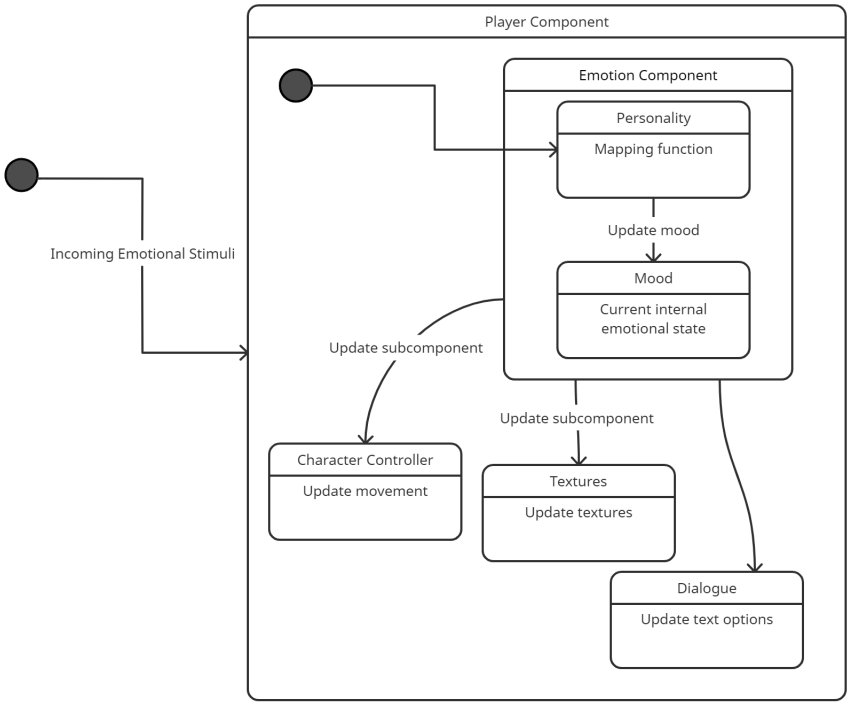

Figure 3.3: An example of a Player state chart with the emotion component

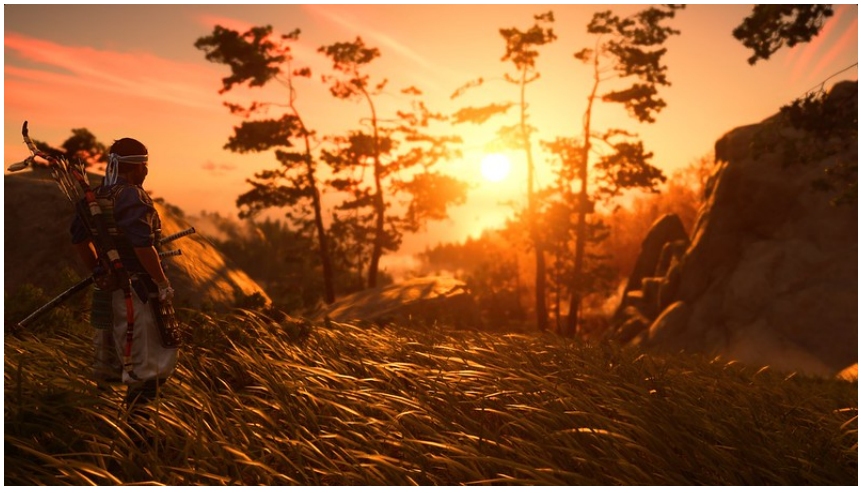

Figure 3.4: The weather system in Ghost of Tsushima changes depending on the player's style [80]. (Image taken from [35]

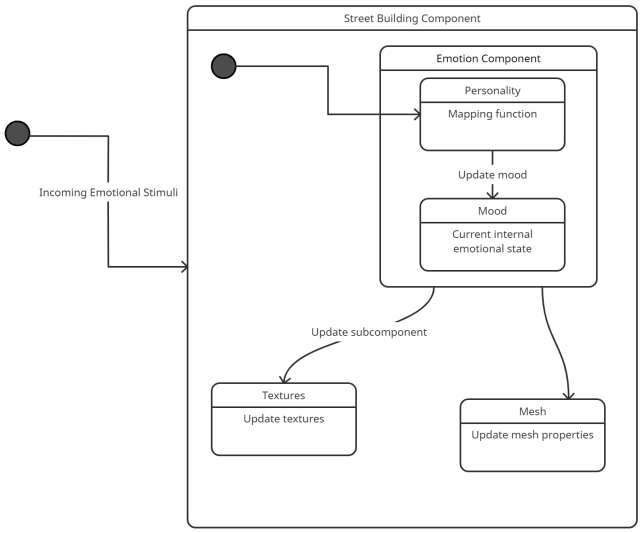

Figure 3.5: An example of a 3D Building Object state chart with the emotion component

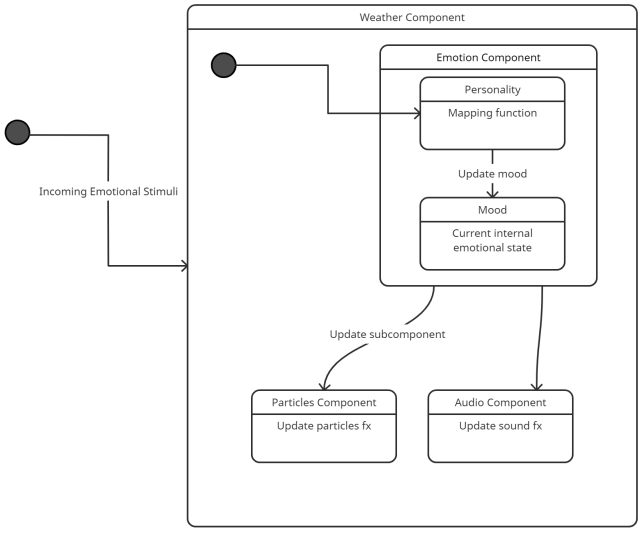

Figure 3.6: An example of a Weather Component state chart with the emotion component

3.4 The Emotion Package

Now that we have set up the precedence of a computational representation of an emotion state, we can now define and describe the architecture of our emotion engine framework. For that, we have to define the emotion subcomponent that will be attached to each object of the game.

The emotion package, seen in Figure 3.7 as attached components to various game objects, consists of three major components. The *personality function*, which describes the core personality of that object. The *mood*, which represents the current emotional state of that object. The *emotion model*, which is the base class of the emotion package.

This emotion package will act as an interface and each component will then have to implement the interface. The emotion engine will in turn trigger each and every component that implements the emotion package interface. This is makes it easier to apply the emotion package to any object.

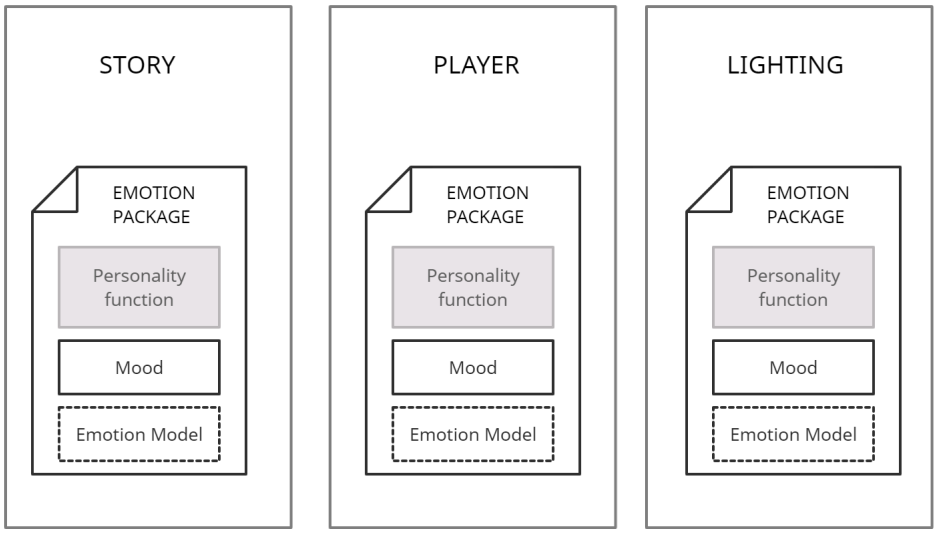

Figure 3.7: The Emotion Package attached to various game objects

3.5 Conceptual Architectural Framework for the Emotion Engine

Now that we have defined and described all the necessary components, we can start designing and building the needed components for the framework. Figure 3.8 shows a

block diagram of all of the different components of the framework.

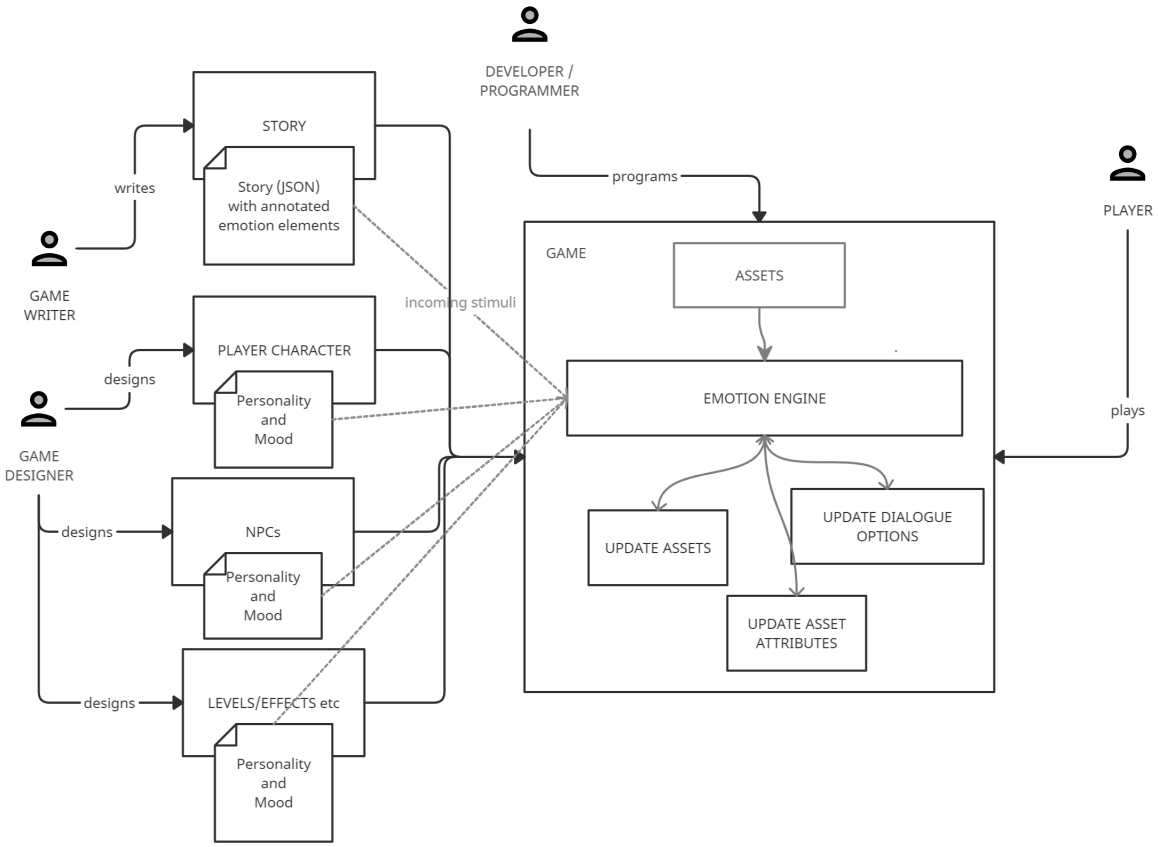

Figure 3.8: Conceptual Framework and Architecture (Overview)

The game designer will design the personality and the current mood of the main player character. Keep in mind that this is not the actual human player but rather the avatar that the human player interacts with. This will include laying out the foundations of the player avatar, their personality, their quirks and their overall personal background. The game designer also designs the personality and the current mood of all Non-Playable Characters (NPCs), if there are any. The designer is free to utilize any theory of emotion. They can also design the personality and mood of all other assets a well, like levels, particles effects etc. As mentioned before, the personality of inorganic objects and agents will just be a function call to update some arbitrary attributes of the object, whereas for human avatars it would be more akin to changing their actual emotional mood.

The story will be written either by a game writer or by the designer. The story is written in a structured and annotated form so as to give each plot point of the story a specific emotion. This means that each major or minor plot point of the story can have an emotional state attached to it. This emotional state will will act as an external emotional stimulus for the various objects within the player avatar's surroundings. The writer and designer can also design quests, or mini missions, and those quests can have emotional states as well. For such a structured document, JSON was chosen as the default format to write the story and the emotional beats of the story. This also gives the writer to write dialogues for NPCs.

The game programmer will then take all assets and the story and finally implement and develop the game. The emotion engine in Figure 3.8 is covered in detail Figure 3.9. Each object that has an emotion component attached to it, the emotion engine takes it and treats it as the current context of that object. The context consists of the mood and the personality attributes of the object. The emotion stimulus E is an incoming stimuli from, say, the story or some other object that triggers a stimulus. It then takes the stimulus, the mood and the personality from the context and then combines them to give a new mood. It does this by calling the personality mapping function and uses the incoming stimulus E to update the internal mood of the context. It then returns the context to the original object, updating it's attributes.

The emotion engine can also take multiple contexts in the form of an array Figure 3.10. This can enable updating multiple objects at the same and can run a single incoming stiumlus E on each context's personality mapping function, giving new mood for their respective internal states.

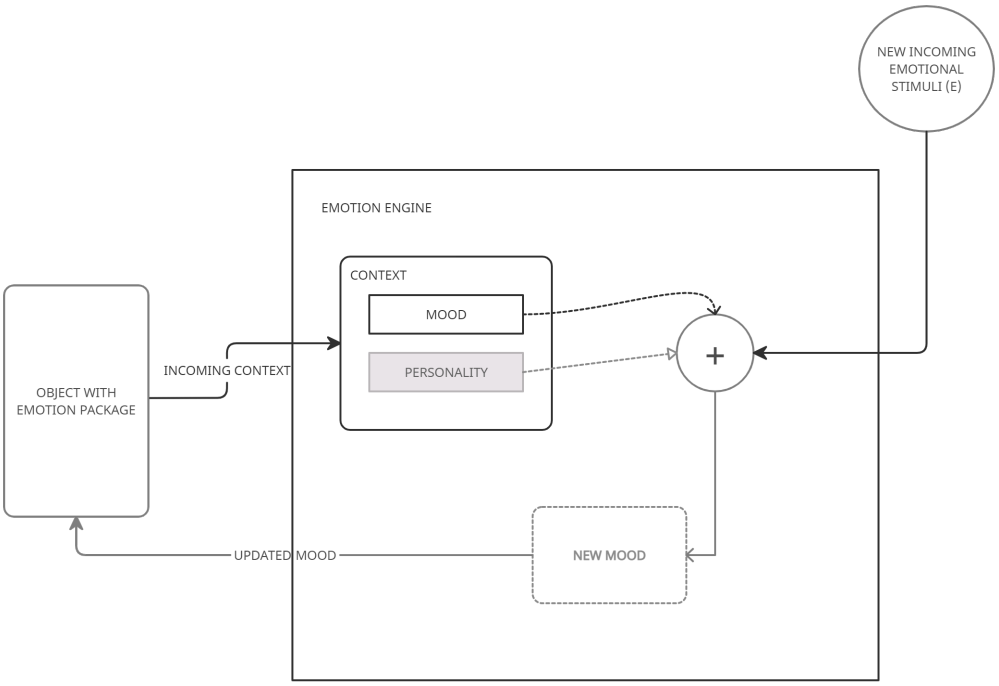

Figure 3.9: Inside View of the Emotion Engine

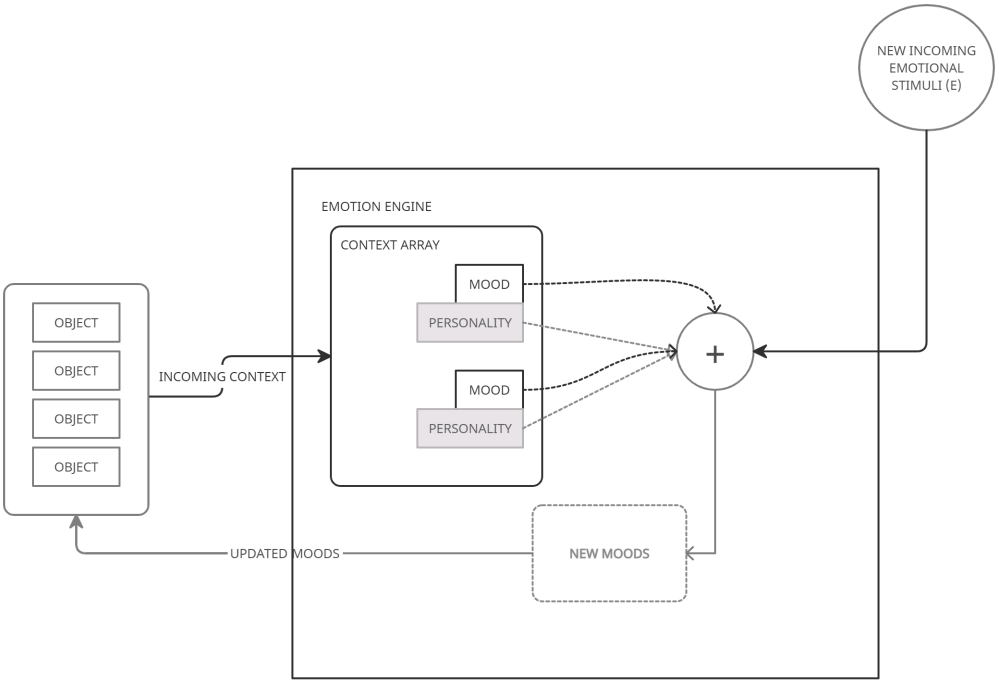

Figure 3.10: Inside View of the Emotion Engine (multiple contexts)

4 Prototype

If a picture is worth a thousand words, a prototype is worth a thousand meetings

IDEO.org

Here we shall discuss the prototypical implementation of the emotion engine framework using the game genre described in section 3.1.1.

4.1 Tools and Technologies Used

For implementing the prototype, certain tools and technologies were selected. Specifically, web technologies were used as they were the most convenient. However, as in the upcoming sections will show, the prototype can be implemented in any programming language, tool and technology.

For the purpose of this prototype, a component-based approach was used. Angular was chosen as the web library to develop the front-end and the user-interface of the video game. Angular is an open-source web platform used to develop SPA (Single Page Applications) for the web. It was also chosen since the framework follows a structured approach to developing components. Angular is also free and easy to learn.

Because of its structured approach, Angular utilizes Typescript as the programming language. The major advantage of Typescript is supports static-type definitions for objects.

A ready-made dashboard template was used to save time on the boiler-plate code for setting up the front-end. In this case, it was CreativeIT's Material Angular Admin Template [57]. It was used because of its dark themes aesthetic and because of its open-source license.

Nodejs was used as the run-time environment and NPM (Node Package Manager) was used to manage the application dependencies.

4.2 Story And Map File

Since the game prototype relies heavily on narrative and exploration as well, to make it easier for the player in our text based adventure game, we utilize a map to show where the player currently is and what possible paths they can take from that position. This will make it easier to write story plot points and descriptive text for each specific location and also annotate each map point with an appropriate emotional beat.

4.2.1 The Map File

A JSON file was used to design the map and annotate each path point. Each path point consists of the x-y coordinates where the player can move to and the pathways extending from that map point. Listing 4.1 shows the listing of the JSON file used to describe the navigational path for the map in Figure 4.1.

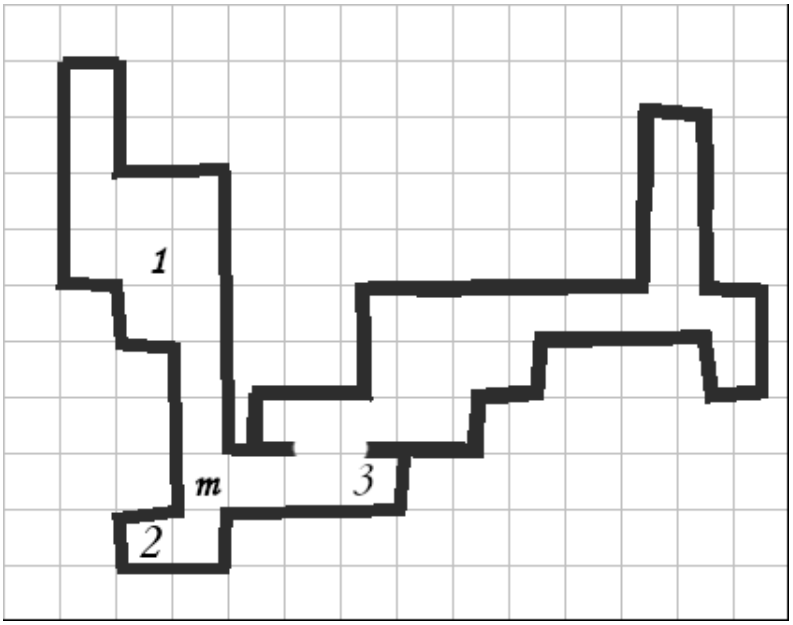

Figure 4.1: Map of the Game, based on The Eagle's Landing [99]

```
{
    "mapPoints": [
        {
            "currentPosition": 0,
            "x": 0,
```

```
            "y": 0,
            "pathsFromCurrentPosition": [
                {
                    "direction": "s",
                    "pathPosition": 1,
                    "x": 50,
                    "y": 100
                }
            ]
        },
        {
            "currentPosition": 1,
            "x": 50,
            "y": 100,
            "pathsFromCurrentPosition": [
                {
                    "direction": "n",
                    "pathPosition": 0,
                    "x": 0,
                    "y": 0
                },
                {
                    "direction": "s",
                    "pathPosition": 2,
                    "x": 53,
                    "y": 210
                }
            ]
        },
        {
            "currentPosition": 2,
            "x": 53,
            "y": 210,
            "pathsFromCurrentPosition": [
                {
                    "direction": "n",
                    "pathPosition": 1,
                    "x": 50,
                    "y": 100
```

```
                },
                {
                    "direction": "s",
                    "pathPosition": 3,
                    "x": 30,
                    "y": 240
                },
                {
                    "direction": "e",
                    "pathPosition": 4,
                    "x": 140,
                    "y": 210
                }
            ]
        },
        {
            "currentPosition": 3,
            "x": 30,
            "y": 240,
            "pathsFromCurrentPosition": [
                {
                    "direction": "n",
                    "pathPosition": 2,
                    "x": 53,
                    "y": 210
                }
            ]
        },
        {
            "currentPosition": 4,
            "x": 140,
            "y": 210,
            "pathsFromCurrentPosition": [
                {
                    "direction": "w",
                    "pathPosition": 2,
                    "x": 53,
                    "y": 210
                }
```

```
            ]
        }
    ]
}
```

Listing 4.1: The Map JSON File

One thing to note about this map JSON file is that designers and programmers can add more map locations to this point and from there can also add extending pathways to other map locations as well. The scalability is limited to the imagination of the designer.

4.2.2 The Story Description File

Narratives for text-based adventure games are written in a unique way, in such that each map location has a unique description text attached to it that shows the player where they are, how the area is and what are the possible pathways extending from there. It is here that the designer and writer can come up with unique and creative ways to describe each map location. Listing 4.2 shows an example of a Story Description File.

```

[
    {
        "mapLocation": 0,
        "text": "You are near the mouth of the cave. Right beside you is
    a worn out wooden sign that reads 'The Cave of Doom'. Darkness emits
    from inside, welcoming only the brave and the courageous.",
        "dialogue": [],
        "stimuli": {
            "anger": 0.0,
            "fear": 0.5,
            "happiness": 0.4,
            "sadness": 0.0
        }
    },
    {
        "mapLocation": 1,
        "text": "The light from the mouth of the cave fades with each
    step you take. Before you know, you are in total darkness. You wait
    until your eyes adjust to the darkness. and you see COLOURS, RAINBOWS,
     FLUFFY BUNNIES. It's EUPHORIA!",
```

```
17          "dialogue": [],
18          "stimuli": {
19              "anger": 0,
20              "fear": 0,
21              "happiness": 1,
22              "sadness": 0.5
23          }
24      },
25      {
26          "mapLocation": 2,
27          "text": "You are now deep into the cave. The pathway splits up
    into two areas.",
28          "dialogue": [],
29          "stimuli": {
30              "anger": 0.0,
31              "fear": 0.7,
32              "happiness": 0,
33              "sadness": 0.2
34          }
35      },
36      {
37          "mapLocation": 3,
38          "text": "You see a TERRIFIED weary old man there.'Greetings,
    traveller!', he says. 'What brings you here?'",
39          "dialogue": [
40              {
41                  "option": 0,
42                  "txt": "'Are you here to help?', the haggard old man asks,
     eyeing you with grey beady eyes.",
43                  "stimuli": {
44                      "anger": 0,
45                      "fear": 0,
46                      "happiness": 0,
47                      "sadness": 0
48                  }
49              },
50              {
51                  "option": 1,
```

```
52                "txt": "'Let me help you!', he says. 'Go here and there.'
    ",
53                "stimuli": {
54                    "anger": 0,
55                    "fear": 0,
56                    "happiness": 1,
57                    "sadness": 0
58                }
59            },
60            {
61                "option": 2,
62                "txt": "'You are here to rob us, aren't you!?', he
    bellows. 'Begone!!'",
63                "stimuli": {
64                    "anger": 1,
65                    "fear": 0,
66                    "happiness": 0,
67                    "sadness": 0
68                }
69            }
70        ],
71        "stimuli": {
72            "anger": 0,
73            "fear": 0,
74            "happiness": 0,
75            "sadness": 0
76        }
77    },
78    {
79        "mapLocation": 4,
80        "text": "You get lost in the corridors.",
81        "dialogue": []
82    }
83 ]
```

Listing 4.2: The Story Description File

Note that each description references the map point by their `id` and each location has an emotional stimuli attached to it. In this particular listing (Listing 4.2), floating point values are used to convey the emotional stimuli each map location has. On line

39, there is also an additional attribute "dialogue", which means that there is an NPC here and can be interacted with. From lines 41 to 67, dialogue options can be seen and even those have emotional stimuli attached to them. Keep in mind that these emotional stimuli are based on the context of the game the designer has in mind; these can easily be changed and are limited to the imagination of the designer and the writer.

4.3 The Emotion Engine Package

The Emotion Engine package is the main package component that provides interfaces, types, the personality function and the emotion engine service.

As discussed in section 3.5, the emotion engine is responsible for calling the mood and the personality function of each object in the context. There is some structure, however, that we need to follow and some rules that we need to set.

First off, the emotion engine service does not need to know much about the details of the context. This means that there should be next to zero dependency or knowledge of about that context; the service should just take that context and call it's personality function and provide the incoming emotional stimulus as a function parameter.

The emotion engine package is composed of the following components:

4.3.1 Emotion Data Type

For this prototype, Ekman's discrete emotions [24] are being used (in a limited capacity). So the `Emotion` type consists of the attributes *sadness*, *anger*, *happiness* and *fear*. Each attribute is of the primitive type `number`. In this case, floating point values are used instead of integers. Listing 4.3 shows the custom Emotion data type.

```
1 export interface Emotion {
2   sadness: number;
3   anger: number;
4   happiness: number;
5   fear: number;
6 }
```

Listing 4.3: The Emotion Data Type (emotion.ts)

4.3.2 Mood

The `Mood` is just a variable of the Emotion type encapsulated as an attribute of an object. Listing 4.4 shows the Mood interface type.

```
1 import { Emotion } from './emotion';
2
3 export interface Mood {
4   emotion: Emotion;
5 }
```

Listing 4.4: Mood (mood.ts)

4.3.3 The IEmotion Interface

To let the emotion engine service know which objects need to have their personality functions called, an interface called IEmotion is used. This interface can be attached to any object (player or non-player) that needs to have an emotional context. This interface will enforce that every object that `extends` from this interface need to implement two components: `currentMood` and the `personality` function. The `personality` function takes in emotional `stimuli` that will "affect" the personality of the agent, and give a new emotion. This emotion will then be set as the `currentMood`.

```
1 import { Emotion } from './emotion';
2
3 export interface IEmotion {
4   currentMood: Emotion;
5   personality: (stimuli: Emotion) => Emotion;
6 }
```

Listing 4.5: The IEmotion Interface (iemotion.ts)

4.3.4 The Emotion Engine affects() Function

The Emotion Engine has only the `affects()` function that takes in two arguments: the current emotional context and an incoming emotional stimulus. The context is the object that implements the emotion interface with its mood and personality mapping function. The incoming stimulus, on the other hand, is from the story emotional beats or from dialogue options.

For an asynchronous approach, the `affects()` function is implemented as an `Observable`, returning an Observable of the type `Emotion`. The emotion engine service then subscribes to the `affects()` function and receives a new emotion. This new emotion is then set as the current mood of the agent. Listing 4.6 shows the emotion engine service and the `affects()` function. Listing 4.7 shows the context data type.

```
1 import { Injectable } from '@angular/core';
```

```
import { Observable, of } from 'rxjs';
import { Context } from './types/context';
import { Emotion } from './types/emotion';

@Injectable({
  providedIn: 'root',
})
export class EmotionEngineService {

  constructor() { }

  /**
   * @param context The current mood and the personality mapping function
     of the object
   * @param newStimuli Incoming affective emotional stimuli
   */
  affects(context: Context, newStimuli: Emotion): Observable<Emotion> {
    const emotion = context.personality(newStimuli);
    return of(emotion);
  }
}
```

Listing 4.6: The Emotion Engine Service (emotion-engine.service.ts)

```
import { Emotion } from './emotion';

export interface Context {
  currentMood: Emotion;
  personality: (stimuli: Emotion) => Emotion;
}
```

Listing 4.7: The Context data type (context.ts)

4.4 Character, Player and the NPC Class

For this game prototype, one player and one NPC was chosen to be implemented. Since both the player avatar and the NPC avatar will be human characters, a base `Character` abstract class was created, which both the `Player` and the `NPC` character classes will inherit. Since both `Player` and the `NPC` character classes will have an

Emotion component attached to them, it was decided to extend the base `Character` class with the `IEmotion` interface. The inherited attributes will then be implemented in the sub-classes. Listing 4.8 shows the source code for the base Character class.

```
1  import { Emotion } from './types/emotion';
2  import { IEmotion } from './types/iemotion';
3
4  abstract class Character implements IEmotion {
5      /** The current mood of the character */
6    currentMood: Emotion;
7
8    constructor();
9    constructor(intialMood: Emotion);
10
11     /**
12      * Constructor and initialize the mood
13      */
14   constructor(initialMood?: Emotion) {
15     this.currentMood = initialMood;
16   }
17
18     /**
19      * @param stimuli The external stimuli that will update the mood of
       the characters
20      */
21   public personality(stimuli: Emotion): Emotion {
22     return this.currentMood;
23   }
24 }
25 export default Character;
```

Listing 4.8: The Base Character Abstract Class (Character.ts)

The `Player` class will then implement the `currentMood` and `personality()` function. Listing 4.9 shows the implementation of `affects()` function in the `Player` class.

```
1 /* @param stimuli The external stimuli that will update the mood of the
     character */
2 public personality(stimuli: Emotion): Emotion {
3     if (stimuli.anger === 0 &&
4             stimuli.fear === 0 &&
```

```
5             stimuli.happiness === 0 &&
6             stimuli.sadness === 0) {
7       Player._setPlayerVisualAttributes(this);
8       return this.currentMood;
9     }
10
11    Player._personalityFunction(stimuli, this);
12
13    return this.currentMood;
14  }
```

Listing 4.9: The Player Personality Function (Player.ts)

Line 11 is a private function call where the emotional stimuli is sent as an argument. The reason a separate function was used because of readability and maintainability. Listing 4.10 shows the details of the private function.

The `personality` function is where personality of the player avatar is described. It is up to the designer and the writer to come up with the description of how the avatar will react when experiencing a certain emotion.

In the prototype implementation in Listing 4.10, for fun, the Player avatar character is described as having mood incongruence; a tough-as-nails barbaric warrior, where he thrives on the fear and sadness of the environment and of other characters, but becomes fearful and sad when there is happiness around him.

```
1  private _personalityFunction(stimuli: Emotion, player: Player) {
2      if (stimuli.happiness >= 0.5) {
3        player.currentMood.fear = 1;
4        player.currentMood.anger -= 0.05;
5        player.currentMood.happiness -= 0.5;
6        player.currentMood.sadness += 0.02;
7      } else if (stimuli.happiness < 0.5) {
8        player.currentMood.fear -= 0.05;
9        player.currentMood.anger += 0.05;
10       player.currentMood.sadness -= 0.05;
11       player.currentMood.happiness += 0.05;
12     }
13
14     if (stimuli.anger >= 0.5) {
15       player.currentMood.happiness += 0.3;
16       player.currentMood.anger += 0.05;
17       player.currentMood.fear -= 0.1;
```

```
18        player.currentMood.sadness -= 0.1;
19      } else if (stimuli.anger < 0.5) {
20        player.currentMood.sadness += 0.3;
21        player.currentMood.fear -= 0.05;
22        player.currentMood.anger += 0.05;
23        player.currentMood.happiness += 0.05;
24      }
25
26      if (stimuli.sadness >= 0.5) {
27        player.currentMood.sadness += 0.05;
28        player.currentMood.happiness += 0.1;
29        player.currentMood.fear -= 0.5;
30        player.currentMood.anger -= 0.05;
31      } else if (stimuli.sadness < 0.5) {
32        player.currentMood.sadness += 0.05;
33        player.currentMood.fear -= 0.05;
34        player.currentMood.anger -= 0.05;
35        player.currentMood.happiness += 0.05;
36      }
37
38      if (stimuli.fear >= 0.5) {
39        player.currentMood.anger += 0.1;
40        player.currentMood.happiness += 0.23;
41        player.currentMood.sadness -= 0.2;
42        player.currentMood.fear = 0;
43      } else if (stimuli.fear < 0.5) {
44        player.currentMood.fear += 0.4;
45        player.currentMood.sadness += 0.05;
46        player.currentMood.anger -= 0.05;
47        player.currentMood.happiness += 0.05;
48      }
49
50      Player._setPlayerVisualAttributes(player);
51    }
```

Listing 4.10: The private Player Personality Function (Player.ts)

Finally, on line 50, another private function is called to set the visual attributes. Here the visual attributes of the player will be updated based on the current mood of the character, thus adapting the textual and visual elements.

Listing 4.10 shows an example of setting various visual attributes of the player character.

```
private static _setPlayerVisualAttributes(player: Player) {
    const maxColourAmt = Math.max(
            player.currentMood.anger,
            player.currentMood.fear,
            player.currentMood.happiness,
            player.currentMood.sadness,
        );

    if (maxColourAmt === player.currentMood.anger) {
      player.moodColour = 'firebrick';
      player.textAnimation = 'shake shake-constant';
      player.moodText = 'Anger';
      if (player.currentMood.anger > 0.7) {
        player.moodText = 'Rage';
      }
    }
    if (maxColourAmt === player.currentMood.happiness) {
      player.moodColour = 'orange';
      player.textAnimation = 'shake-slow shake-constant';
      player.moodText = 'Happiness';
      if (player.currentMood.happiness > 0.7) {
        player.moodText = 'Ecstasy';
      }
    }
    if (maxColourAmt === player.currentMood.sadness) {
      player.moodColour = 'teal';
      player.textAnimation = '';
      player.moodText = 'Sadness';
      if (player.currentMood.sadness > 0.7) {
        player.moodText = 'Depression';
      }
    }
    if (maxColourAmt === player.currentMood.fear) {
      player.moodColour = 'darkgreen';
      player.textAnimation = 'shake-opacity shake-constant';
      player.moodText = 'Fear';
```

```
37      if (player.currentMood.fear > 0.7) {
38        player.moodText = 'Paralysis from Fear';
39      }
40    }
41 }
```

Listing 4.11: The private Player Visual Attributes Function (Player.ts)

Lines 9 to 38 set various attributes of the color properties and textual animation, based on the current mood of the player.

Similarly to the Player, we can also describe the personality of the NPC, shown in Listing 4.12. Here, the NPC was described as having a mood congruent personaltiy, where, for example, whenever he experiences a stimulus that has fearful annotations, he will get fearful as well. To make things simpler, the NPC's current mood is set to whatever the max numeric value of the list of emotion is.

```
1 /* @param stimuli The external stimuli that will update the mood of the
      character */
2 public personality(stimuli: Emotion): Emotion {
3     const maxEmotion = Math.max(
4             stimuli.anger,
5             stimuli.fear,
6             stimuli.happiness,
7             stimuli.sadness,
8         );
9
10    if (maxEmotion === stimuli.anger) {
11      this.moodColour = 'firebrick';
12    }
13    if (maxEmotion === stimuli.happiness) {
14      this.moodColour = 'orange';
15    }
16    if (maxEmotion === stimuli.sadness) {
17      this.moodColour = 'teal';
18    }
19    if (maxEmotion === stimuli.fear) {
20      this.moodColour = 'darkgreen';
21    }
22    this.currentMood = stimuli;
23    return this.currentMood;
```

```
24 }
```

Listing 4.12: The NPC Personality Function (NPC.ts)

4.5 The Text-Based Game

Figure 4.2 shows an overview of the text-based adventure game. It consists of the following components:

4.5.1 Map

The top most area is the *Map* component, where the map image of the level can be seen. The grey circle at the top is the player character while there is another colored circle at the bottom of the map, which is the NPC. The locations are numbered *1, 2, 3* and *m*, which shows a multiple-path position. To the right of the map image is the story and the map location description. This is the same text in Listing 4.1 that is parsed from the JSON file.

Players can navigate around the map using the arrows at the bottom Player area. The map follows the layout very strictly; players can not go where there are walls.

4.5.2 Player

The Player component on the bottom left shows the visual elements associated with a particular emotion that the player avatar is feeling. Depending on the current mood, a unique text animation would be assigned to the text showing the current mood. These can be seen in the variable `textAnimation` from lines 9 to 38 in Listing 4.11. From the same listing and the same line numbers, the variable `moodColour`, used for different colors for different emotions, and `moodText`, are also set depending on the current mood.

The Player component also has navigation arrows, for allowing the player to move either top, left, bottom or right.

4.5.3 Mood

The Mood component on the bottom right shows the current emotion of the player character avatar. For this prototype, there are four emotions associated with each mood, namely *happiness*, *fear*, *sadness* and *anger*. The choice of the colors used for each emotion is based on Plutchik's Wheel of Emotion in Figure 2.1, but other visual aesthetics can be used as well based on symbolism and context.

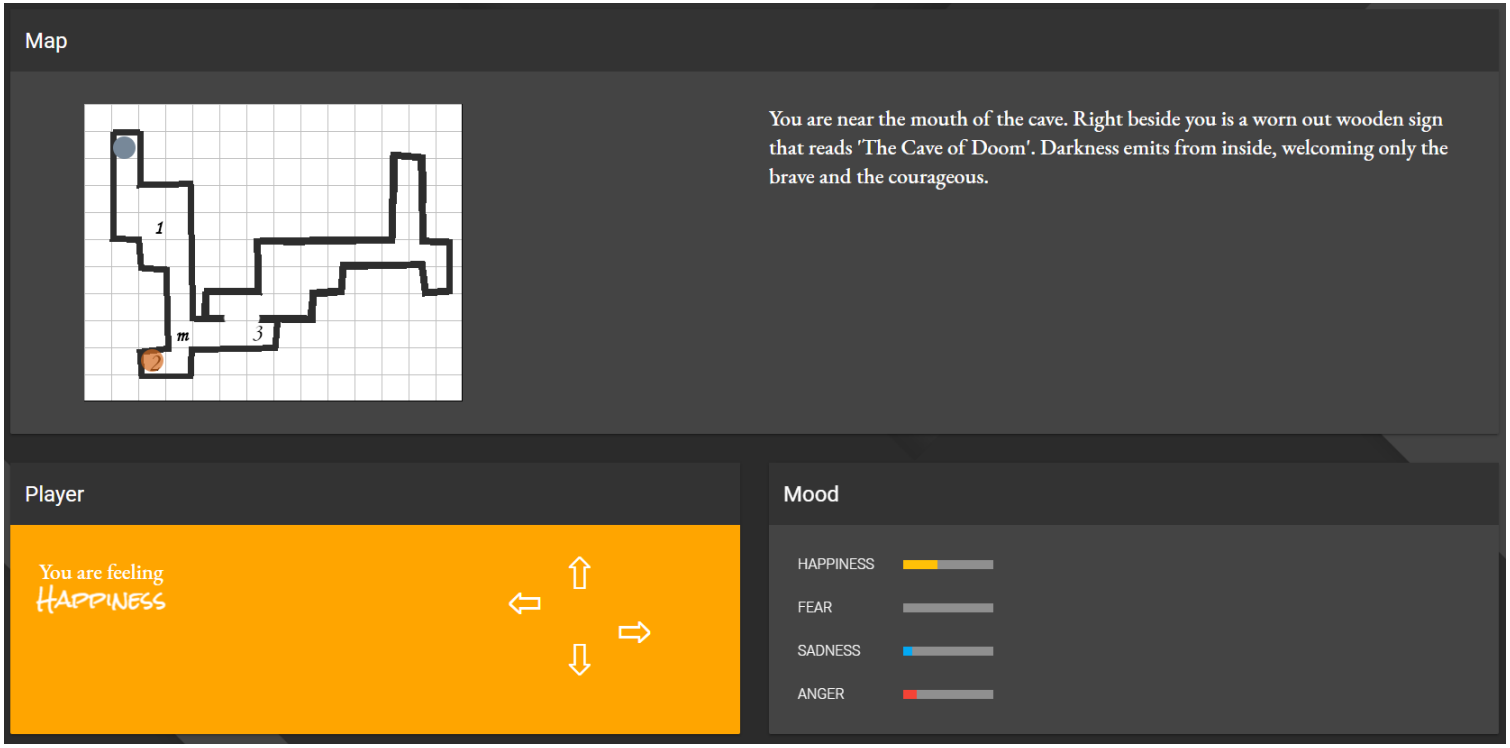

Figure 4.2: The Text-Based Game Prototype

The Player component also has navigation arrows, for allowing the player to head either top, left, bottom or right.

4.5.4 Gameplay

The gameplay consists of reading the story text and navigating the player avatar through clicking the arrow keys. Each map location has an emotion element attached to it, from the JSON listings in Listing 4.1. The emotion element from each map and story point gets thrown into the emotion engine as an incoming external stimuli. The emotion engine then takes the personality of the player avatar and calls the personality function and returns an updated mood. The mood gets then reflected back to the player via visual elements, such as animation and color (see Figure 4.3).

The player avatar can also interact with NPCs as well (shown in Figure 4.4). Depending on the choice the player makes, the NPC's internal mood changes as well. In this prototype, since the player avatar is a mood incongruent brutal warrior, saying 'no' to the NPC (Figure 4.5) makes the NPC angry at the avatar. This in turn makes the player avatar supremely happy (shown in Figure 4.6). The NPC circle also changes from a orange neutral to red anger.

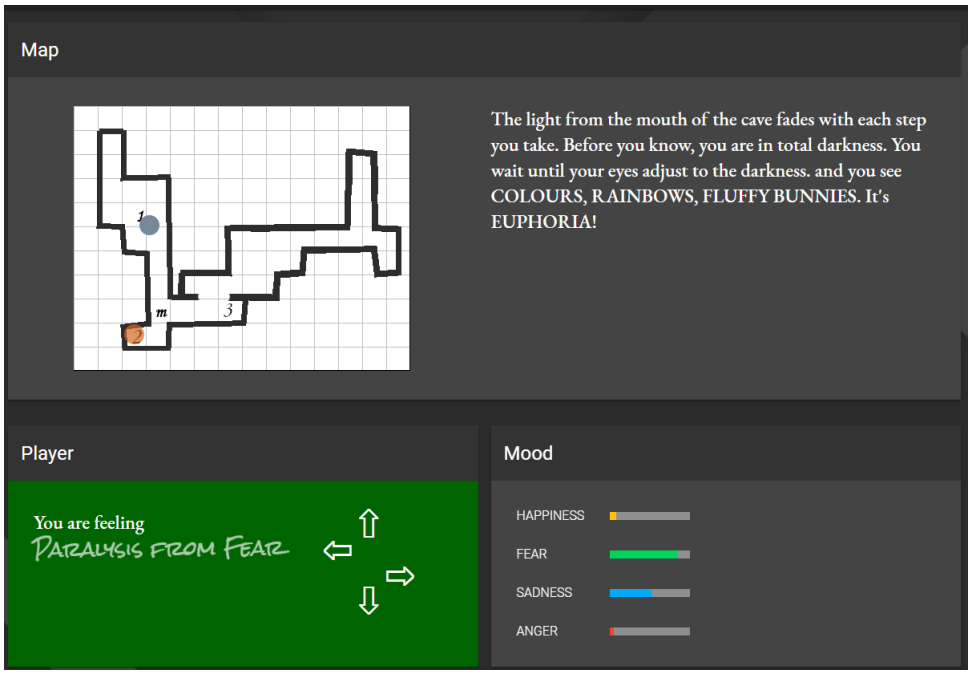

Figure 4.3: The Player Avatar's mood changes, from happiness to fear

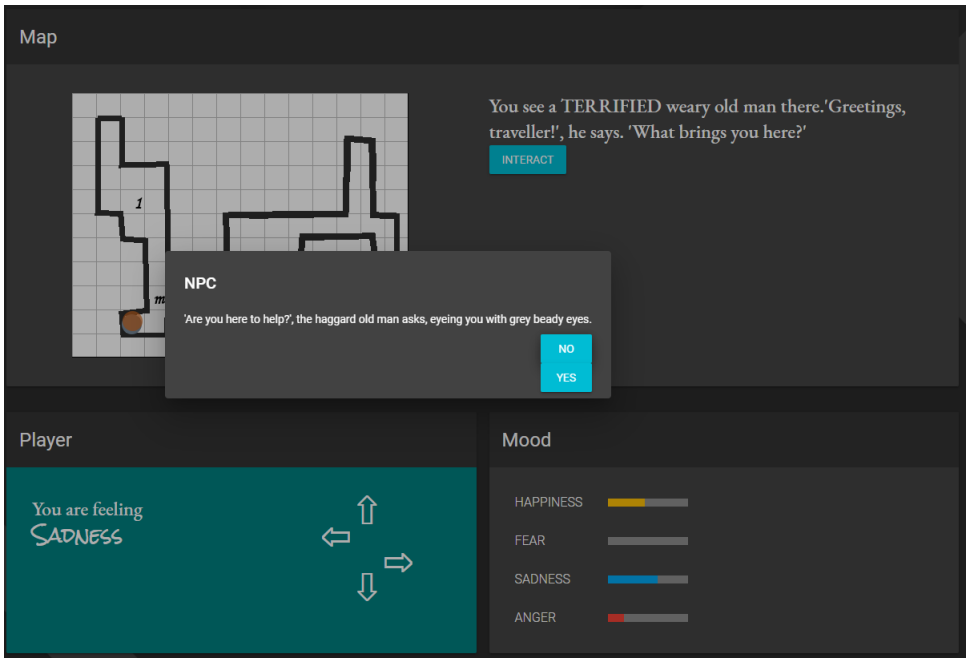

Figure 4.4: The Player Avatar interacting with an NPC

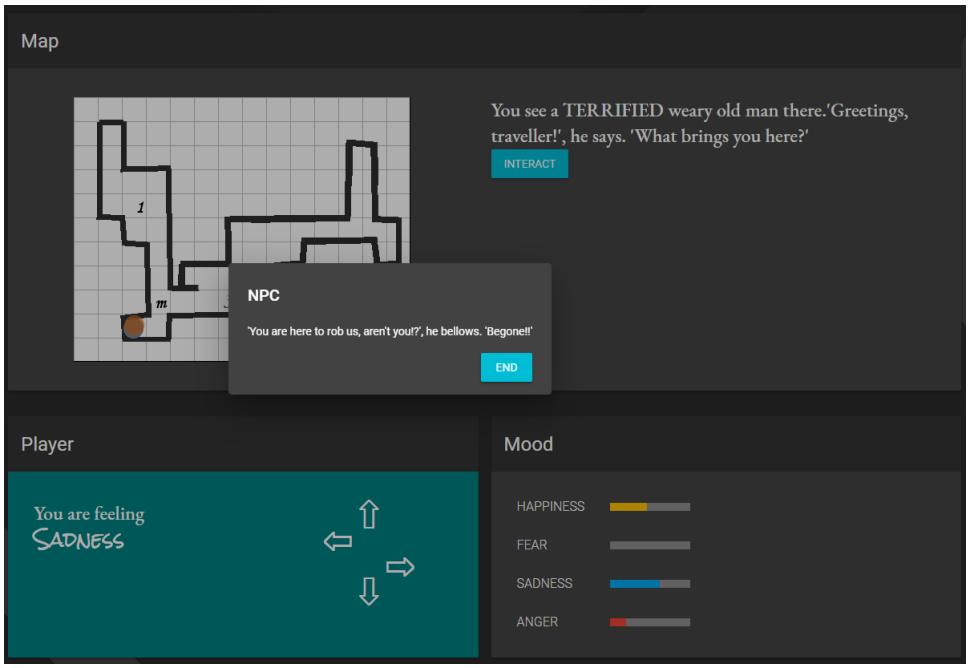

Figure 4.5: The NPC, angry at the Player avatar, results in both characters' having their moods changed.

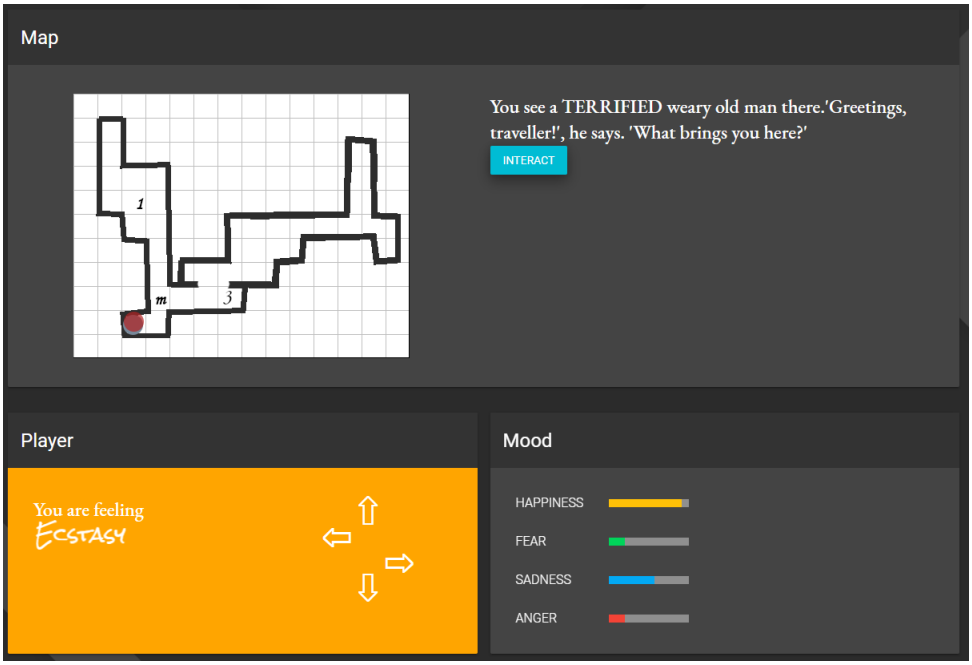

Figure 4.6: The Player Avatar and the NPC mood changes

5 Evaluation

Cogito, ergo sum

René Descartes

To evaluate the system and its prototype (described in chapter 3 and chapter 4 respectively), comparisons need to be made with other similar systems described in chapter 2.

5.1 Evaluation Criteria

The evaluation is done based on six criteria (see Table 5.1). The *type* determines the technical categorization of the system. GAMGYDALA[77], for example, was developed as a usable library, whereas other frameworks like ERiSA[17] and PaSSAGE[100] are more conceptual and need to be either implemented or researched upon. Our proposed framework, like others, needs to be implemented first in the tool of our choice. *Adaptive Narrative* determines whether the system adapts story and plot points or not and how much. PaSSAGE and PACE[33] are quite capabale of having an adaptive narrative, while our framework is limited. *Adaptive Visual Elements* determines whether the system has the capabilites to adapt visual elements (such as colors, camera settings, audio settings) based on the player's playstyle or not. As seen from Table 5.1, only two are capable: one is the Affective Narrative Framework [28] and the current proposed framework. Even so, the former is quite limited than the latter. *NPC Emotions* determine whether the system has capabilities to apply emotional models to Non-Playable Characters. As before, our system is quite limited. *Player Modeling and Influence* explores whether the system can be influenced by the human player or whether the system tried to model the player based on their playstyle. GAMYGDALA and ERiSA are quite limited in this aspect, since they are more focused on adding emotional elements to NPCs rather than the player avatar. PaSSAGE and PACE, on the other hand, actively attempt to model the player during gameplay. The proposed framework, however, works the other way around: it influences the player using its adaptive game narrative. Finally, *usage* determines how the system can be used. GAMYGDALA is the only system in Table 5.1 that can be used out-of-the-box as other systems, including our own, need to implemented first.

Table 5.1: Comparison with other systems

Category	**GAMYGDALA [77]**	**ERiSA [17]**	**PaSSAGE [100]**	**PACE [34]**	**Affective Narrative Framework [28]**	***Proposed Framework***
Type	Usable library	Conceptual Framework	A.I. based System	A.I. based System	Implementable Framework	Implementable Framework
Adaptive Narrative	No	No	Yes	Yes	Yes (limited)	Yes (limited)
Adaptive Visual Elements	No	No	No	No	Yes (limited)	Yes
NPC Emotions	Extensive	Extensive	Limited but dynamic	Limited but dynamic	Limited	Limited
Player Modeling and Influence	Limited	Limited	Models player extensively for interactive storytelling	Player influences game narrative and NPCs	Player influences game narrative and NPCs	Game narrative influences player and NPCs
Usage	Out-of-the-box and as a plug-in	Conceptual	Conceptual	Conceptual and Algorithmic	Framework needs to be implemented	Framework needs to be implemented

6 Conclusion and Outlook

Roads? Where we're going we don't need roads.

Dr. Emmett Brown (Back to the Future)

Video games are a profound medium when it comes to interactive storytelling. The interactive nature of this medium provides an unprecedented immersive experience to the players. They can choose their own playstyles and make choices for their avatar and experience the same story differently on multiple playthroughs. Apart from the story, since modern video games have now budgets that parallel big-budget summer blockbuster movies, their visuals have significantly improved as well, and they look more and more photo-realistic with each passing year.

Story and visual elements are thus quite important when it comes to considering elements that can be adapted to each player's playstyle. For that, a framework was needed to explore this idea of adapting narrative and visual elements of a video game.

To develop this framework, much research was done on related topics. We started out from understanding the biological and psychological aspects of emotions. After that, we explored some computational models of emotions, which techniques each one used and what their advantages and shortcomings were. Since we are more focused on system design, we discussed the computational models of emotion within the context of software engineering. We needed a formal methodology to discuss these computational models, since most of the models were developed within the context of scientific research and thus do not have a formal methodology. There, we explored various emotion engines that followed software engineering methodologies to some extent. Moving on, we discussed aesthetic emotions to give arguments to adapt not only story elements but also other visual elements. Finally, to begin development, we tied everything together and discussed video games and adaptive narratives.

After this research, we came up with a concept for a framework. We implemented it in an appropriate tool and developed a small prototype. We chose a specific game genre and wrote a small narrative for it. We then evaluated the framework and compared it to other frameworks and emotion engines.

From those comparisons, we saw some shortcomings and also some potential improvements that can be done in the future.

6.1 Shortcomings

Availability Our proposed framework should work right out of the box as either a plugin or a standalone tool that can be integrated with any other tool.

Character Personalities When it comes to designing personalities of NPCs (Non-Playable Characters), there is still some limitation as to how they are designed in our system. Other systems either employ complex mathematical models or some other distinguishing feature to handle personalities, whereas our system is dependent on the game designer.

Genre Even though the proposed framework should theoretically work with any genre, implementing the framework for a game with a different genre (such as action adventure) warrants a trial.

Visuals A follow up to the previous shortcoming, the proposed framework was used to develop a prototype for a text-based adventure game. The visual aspects of the game are quite basic, with basic primitive shapes like circles representing the characters. Usage of this framework within a 3D game warrants a trial.

6.2 Future Work

Plugin or Standalone App or Service The proposed framework in its current state needs to be implemented before it can be used. However, the framework can also be developed as a plugin for popular game engines (such as Unity and Unreal Engine), providing easy integration. Or it can be developed as a standalone desktop app or hosted online and be used as a web service.

Complex Models of Emotion The proposed framework uses a simplistic model of emotion. Future work can also explore adapting visual elements while using complex models of emotion (such as the OCC or any other appraisal based model).

Advanced Prototypes Another aspect that can be looked in the future is the development of advanced prototypes. Games in different genres and different visual aspects can be developed and tested to see if the framework works with those or not.

User Testing A follow-up to the previous point, user testing should be carried out and their results should be evaluated based on some criteria.

List of Figures

List of Tables

Listings

Ludography

[1] *80 Days*. Inkle, 2014.

[25] *Fallout 4*. Bethesda Game Studios, 2015.

[81] *Red Dead Redemption*. Rockstar Games, 2010.

[102] *Uncharted: Drake's Fortune*. Naughty Dog, 2007.

[105] *Zork*. Infocom, 1977.

Bibliography

[2] H. P. Abbott. *The Cambridge introduction to narrative*. Cambridge University Press, 2020.

[3] E. W. Adams. "Will computer games ever be a legitimate art form?" In: *International Journal of Technology Management & Sustainable Development* 7.1 (2008), pp. 67–77.

[4] E. Adams and A. Rollings. *Fundamentals of game design (game design and development series)*. Prentice-Hall, Inc., 2006.

[5] C. Becker-Asano and I. Wachsmuth. "Affective computing with primary and secondary emotions in a virtual human." In: *Autonomous Agents and Multi-Agent Systems* 20.1 (2010), pp. 32–49.

[6] A. Behrad. *Hippocampus and Memory Consolidation: Parallel System Theory*.

[7] D. E. Berlyne. *Aesthetics and psychobiology*. 1973.

[8] H. Bouazza and F. Bendella. "Adaptation of a model of emotion regulation to modulate the negative emotions based on persistency." In: *Multiagent and Grid Systems* 13.1 (2017), pp. 19–30.

[9] E. J. Braude and M. E. Bernstein. *Software engineering: modern approaches*. Waveland Press, 2016.

[10] J. Broekens, T. Bosse, and S. C. Marsella. "Challenges in computational modeling of affective processes." In: *IEEE Transactions on Affective Computing* 4.3 (2013), pp. 242–245.

[11] J. Broekens, D. DeGroot, and W. A. Kosters. "Formal models of appraisal: Theory, specification, and computational model." In: *Cognitive Systems Research* 9.3 (2008), pp. 173–197.

[12] B. Bruegge and A. H. Dutoit. "Object–Oriented Software Engineering. Using UML, Patterns, and Java." In: *Learning* 5.6 (2009), p. 7.

[13] W. B. Cannon. "The James-Lange theory of emotions: A critical examination and an alternative theory." In: *The American journal of psychology* 39.1/4 (1927), pp. 106–124.

[14] S. Castellanos, L.-F. Rodrıguez, L. A. Castro, and J. O. Gutierrez-Garcia. "A computational model of emotion assessment influenced by cognition in autonomous agents." In: *Biologically inspired cognitive architectures* 25 (2018), pp. 26–36.

[15] J. Chen. "Flow in games (and everything else)." In: *Communications of the ACM* 50.4 (2007), pp. 31–34.

[16] A. Chowanda, P. Blanchfield, M. Flintham, and M. Valstar. "Computational models of emotion, personality, and social relationships for interactions in games." In: *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. 2016, pp. 1343–1344.

[17] A. Chowanda, P. Blanchfield, M. Flintham, and M. Valstar. "Erisa: Building emotionally realistic social game-agents companions." In: *International conference on intelligent virtual agents*. Springer. 2014, pp. 134–143.

[18] J. Costa and T. Paul. "of personality theories: Theoretical contexts for the five-factor model." In: *The five-factor model of personality: Theoretical perspectives* 51 (1996).

[19] C. Crawford. *The art of computer game design*. Osborne/McGraw-Hill Berkeley, CA, 1984.

[20] M. Csikszentmihalyi. *Flow: The psychology of optimal experience*. Vol. 1990. Harper & Row New York, 1990.

[21] G. C. Cupchik. "Emotion in aesthetics: Reactive and reflective models." In: *Poetics* 23.1-2 (1995), pp. 177–188.

[22] P. De Byl. "A conceptual affective design framework for the use of emotions in computer game design." In: *Cyberpsychology: Journal of Psychosocial Research on Cyberspace* 9.3 (2015).

[23] P. Ekman. "An argument for basic emotions." In: *Cognition & emotion* 6.3-4 (1992), pp. 169–200.

[24] P. Ekman. "Emotions revealed." In: *Bmj* 328.Suppl S5 (2004).

[26] S. Fitch. "From Survival to THRIVAL." In: *Paediatrics & child health* 11.9 (2006), pp. 575–576.

[27] E. M. Forster. "Aspects of the Novel. 1927." In: *London: Edward Arnold* (1974).

[28] M. A. M. D. Freilão. "Affective narratives for engagement in digital games." In: (2020).

[29] N. H. Frijda. "Aesthetic emotions and reality." In: (1989).

[30] A. Gabrielsson and S. L. Wik. "Strong experiences related to music: adescriptive system." In: *Musicae scientiae* 7.2 (2003), pp. 157–217.

[31] S. Gu, F. Wang, N. P. Patel, J. A. Bourgeois, and J. H. Huang. "A model for basic emotions using observations of behavior in Drosophila." In: *Frontiers in psychology* 10 (2019), p. 781.

[32] J. M. Haviland et al. *Handbook of emotions*. Guilford Press, 2000.

[33] S. P. Hernandez, V. Bulitko, and E. S. Hilaire. "Emotion-based interactive storytelling with artificial intelligence." In: *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. Vol. 10. 1. 2014.

[34] S. P. Hernandez, V. Bulitko, and M. Spetch. "Keeping the player on an emotional trajectory in interactive storytelling." In: *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. Vol. 11. 1. 2015.

[35] herraleco. *Ghost of Tsushima 20200719210322*. URL: `https://www.flickr.com/photos/189676329@N08/50195414362/`.

[36] C. Hieida, T. Horii, and T. Nagai. "Deep emotion: A computational model of emotion using deep neural networks." In: *arXiv preprint arXiv:1808.08447* (2018).

[37] R. Houlette. "Player modeling for adaptive games." In: *AI game programming wisdom* (2003), pp. 557–566.

[38] E. Hudlicka. "This time with feeling: Integrated model of trait and state effects on cognition and behavior." In: *Applied Artificial Intelligence* 16.7-8 (2002), pp. 611–641.

[39] N. Ivanov. *Cakechat: Emotional generative dialog system*. 2019. URL: `https://github.com/lukalabs/cakechat#quick-start`.

[40] C. E. Izard. "Basic emotions, relations among emotions, and emotion-cognition relations." In: (1992).

[41] W. James. "The emotions." In: (1922).

[42] H. Jenkins. "Game design as narrative architecture." In: *Computer* 44.3 (2004), pp. 118–130.

[43] I. Kant. *Critique of Judgement*. Macmillan London, 1931.

[44] A. Kołakowska, A. Landowska, M. Szwoch, W. Szwoch, and M. R. Wróbel. "Emotion recognition and its application in software engineering." In: *2013 6th International Conference on Human System Interactions (HSI)*. IEEE. 2013, pp. 532–539.

[45] R. Koster. *Theory of fun for game design*. " O'Reilly Media, Inc.", 2013.

[46] J. Krüger. "Interactive Storytelling and Emotions." In: (2020).

[47] R. S. Lazarus and R. S. Lazarus. *Emotion and adaptation*. Oxford University Press on Demand, 1991.

[48] J. E. LeDoux. "Chapter 21 - Evolution of human emotion: A view through fear." In: Progress in Brain Research 195 (2012). Ed. by M. A. Hofman and D. Falk, pp. 431–442. ISSN: 0079-6123. DOI: https://doi.org/10.1016/B978-0-444-53860-4.00021-0.

[49] J. E. LeDoux. "Emotion circuits in the brain." In: *Annual review of neuroscience* 23.1 (2000), pp. 155–184.

[50] P. D. MacLean. "Contrasting functions of limbic and neocortical systems of the brain and their relevance to psychophysiological aspects of medicine." In: *The American journal of medicine* 25.4 (1958), pp. 611–626.

[51] R. Mall. *Fundamentals of software engineering*. PHI Learning Pvt. Ltd., 2018.

[52] R. A. Mar, K. Oatley, M. Djikic, and J. Mullin. "Emotion and narrative fiction: Interactive influences before, during, and after reading." In: *Cognition & emotion* 25.5 (2011), pp. 818–833.

[53] R. P. Marinier III, J. E. Laird, and R. L. Lewis. "A computational unification of cognitive behavior and emotion." In: *Cognitive Systems Research* 10.1 (2009), pp. 48–69.

[54] S. C. Marsella and J. Gratch. "EMA: A process model of appraisal dynamics." In: *Cognitive Systems Research* 10.1 (2009), pp. 70–90.

[55] S. Marsella and J. Gratch. "Computationally modeling human emotion." In: *Communications of the ACM* 57.12 (2014), pp. 56–67.

[56] S. Marsella, J. Gratch, P. Petta, et al. "Computational models of emotion." In: *A Blueprint for Affective Computing-A sourcebook and manual* 11.1 (2010), pp. 21–46.

[57] *Material Angular Admin Template*. URL: https://github.com/CreativeIT/material-angular-dashboard.

[58] D. Matravers. *Art and emotion*. Oxford University Press, 2001.

[59] R. R. McCrae and P. T. Costa. "Validation of the five-factor model of personality across instruments and observers." In: *Journal of personality and social psychology* 52.1 (1987), p. 81.

[60] R. Misslin. "The defense system of fear: behavior and neurocircuitry." In: *Neurophysiologie Clinique/Clinical Neurophysiology* 33.2 (2003), pp. 55–66. ISSN: 0987-7053. DOI: https://doi.org/10.1016/S0987-7053(03)00009-1.

[61] A. Moors and P. Ellsworth. “Scherer, KR, & Frijda, N. H.(2013).” In: *Appraisal theories of emotion: State of the art and future development. Emotion Review* 5.2 (), pp. 119–124.

[62] J. Nakamura and M. Csikszentmihalyi. “The concept of flow.” In: *Flow and the foundations of positive psychology*. Springer, 2014, pp. 239–263.

[63] M. S. El-Nasr, J. Yen, and T. R. Ioerger. “Flame—fuzzy logic adaptive model of emotions.” In: *Autonomous Agents and Multi-agent systems* 3.3 (2000), pp. 219–257.

[64] S. Ojha and M.-A. Williams. “Ethically-guided emotional responses for social robots: Should i be angry?” In: *International conference on social robotics*. Springer. 2016, pp. 233–242.

[65] S. Ojha, M.-A. Williams, and B. Johnston. “The essence of ethical reasoning in robot-emotion processing.” In: *International Journal of Social Robotics* 10.2 (2018), pp. 211–223.

[66] U. Olsson and I. Å. Svensson. “E. 2005.” In: *Kalkylering för produkter och investeringar* (2005).

[67] A. Ortony, G. L. Clore, and A. Collins. *The cognitive structure of emotions*. Cambridge university press, 1990.

[68] E. Osuna, L.-F. Rodrıguez, J. O. Gutierrez-Garcia, and L. A. Castro. “Development of computational models of emotions: A software engineering perspective.” In: *Cognitive Systems Research* 60 (2020), pp. 1–19.

[69] F. Parker. “An art world for artgames.” In: *Loading...* 7.11 (2013).

[70] F. Peng, V. C. LaBelle, E. C. Yue, and R. W. Picard. “A trip to the moon: Personalized animated movies for self-reflection.” In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 2018, pp. 1–10.

[71] L. Perlovsky. “Emotions of" higher" cognition.” In: *Behavioral and Brain Sciences* 35.3 (2012), p. 157.

[72] L. I. Perlovsky. *Neural networks and intellect: Using model-based concepts*. Vol. 51. Oxford University Press New York, 2001.

[73] L. I. Perlovsky. “Toward physics of the mind: Concepts, emotions, consciousness, and symbols.” In: *Physics of Life Reviews* 3.1 (2006), pp. 23–55.

[74] L. I. Perlovsky, M.-C. Bonniot-Cabanac, and M. Cabanac. “Curiosity and pleasure.” In: *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2010, pp. 1–3.

[75] D. C. Plaut. “Methodologies for the computer modeling of human cognitive processes.” In: *To appear in F. Boller and J. Grafman (Eds.)* (2000).

[76] R. Plutchik. "Emotion." In: *A psychoevolutionary synthesis* (1980).

[77] A. Popescu, J. Broekens, and M. Van Someren. "Gamygdala: An emotion engine for games." In: *IEEE Transactions on Affective Computing* 5.1 (2013), pp. 32–44.

[78] J. Posner, J. A. Russell, and B. S. Peterson. "The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology." In: *Development and psychopathology* 17.3 (2005), p. 715.

[79] R. S. Pressman. *Software engineering: a practitioner's approach*. Palgrave macmillan, 2005.

[80] J. Ramée. *Ghost of Tsushima Weather Changes*. 2020. URL: https://www.gamespot.com/articles/ghost-of-tsushima-changes-its-weather-depending-on/1100-6479889/.

[82] L.-F. Rodrıguez, J. O. Gutierrez-Garcia, and F. Ramos. "Modeling the interaction of emotion and cognition in autonomous agents." In: *Biologically Inspired Cognitive Architectures* 17 (2016), pp. 57–70.

[83] L.-F. Rodrıguez and F. Ramos. "Development of computational models of emotions for autonomous agents: a review." In: *Cognitive Computation* 6.3 (2014), pp. 351–375.

[84] J. Rumbaugh, I. Jacobson, and G. Booch. "The unified modeling language." In: *Reference manual* (1999).

[85] J. A. Russell. "A circumplex model of affect." In: *Journal of personality and social psychology* 39.6 (1980), p. 1161.

[86] J. A. Russell. "Core affect and the psychological construction of emotion." In: *Psychological review* 110.1 (2003), p. 145.

[87] K. R. Scherer. "What are emotions? And how can they be measured?" In: *Social science information* 44.4 (2005), pp. 695–729.

[88] F. Schoeller, L. Perlovsky, and D. Arseniev. "Physics of mind: experimental confirmations of theoretical predictions." In: *Physics of life reviews* 25 (2018), pp. 45–68.

[89] D. Schultz. *Computational model of appraisal*. 2014. URL: https://github.com/derek-schultz/appraisal-model.

[90] M. Sharma, S. Ontañón, M. Mehta, and A. Ram. "Drama management and player modeling for interactive fiction games." In: *Computational Intelligence* 26.2 (2010), pp. 183–211.

[91] P. Shaver, J. Schwartz, D. Kirson, and C. O'connor. "Emotion knowledge: further exploration of a prototype approach." In: *Journal of personality and social psychology* 52.6 (1987), p. 1061.

[92] P. J. Silvia. "Looking past pleasure: anger, confusion, disgust, pride, surprise, and other unusual aesthetic emotions." In: *Psychology of Aesthetics, Creativity, and the Arts* 3.1 (2009), p. 48.

[93] P. J. Silvia. "What is interesting? Exploring the appraisal structure of interest." In: *Emotion* 5.1 (2005), p. 89.

[94] C. A. Smith, R. S. Lazarus, et al. "Emotion and adaptation." In: *Handbook of personality: Theory and research* (1990), pp. 609–637.

[95] I. Sommerville. *Software engineering 9th Edition*. Vol. 137035152. 2011, p. 18.

[96] *Statecharts*. URL: https://statecharts.github.io/what-is-a-statechart.html.

[97] B. R. Steunebrink, M. Dastani, and J.-J. C. Meyer. "The OCC model revisited." In: *Proc. of the 4th Workshop on Emotion and Computing*. Association for the Advancement of Artificial Intelligence. 2009.

[98] R. Taylor et al. "Translator's notes to an article on Babbage's Analytical Engiro." In: *Scientific Memoirs* 3 (1842), pp. 691–731.

[99] *The Eagle's Landing*. URL: https://adventurematerials.files.wordpress.com/2011/07/the-eagles-landing.png?w=640.

[100] D. Thue, V. Bulitko, M. Spetch, and E. Wasylishen. "Interactive Storytelling: A Player Modelling Approach." In: *AIIDE*. 2007, pp. 43–48.

[101] A. M. TURING. "I.—COMPUTING MACHINERY AND INTELLIGENCE." In: *Mind* LIX.236 (Oct. 1950), pp. 433–460. ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433. eprint: https://academic.oup.com/mind/article-pdf/LIX/236/433/30123314/lix-236-433.pdf.

[103] J. D. Velásquez. "Cathexis–a computational model for the generation of emotions and their influence in the behavior of autonomous agents." PhD thesis. Massachusetts Institute of Technology, 1996.

[104] A. J. Zautra, G. G. Affleck, H. Tennen, J. W. Reich, and M. C. Davis. "Dynamic Approaches to Emotions and Stress in Everyday Life: Bolger and Zuckerman Reloaded With Positive as Well as Negative Affects." In: *Journal of Personality* 73.6 (2005), pp. 1511–1538. DOI: https://doi.org/10.1111/j.0022-3506.2005.00357.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.0022-3506.2005.00357.x.