An Agent Framework for a Modular Serious Game

Pauline Jepp
INESC-ID
Lisboa, Portugal
Email: pjepp@inesc-id.pt

Manuel Fradinho
Cyntelix
London, UK
Email: mfradinho@cyntelix.com

João Madeiras Pereira
INESC-ID
Lisboa, Portugal
Email: jap@inesc.pt

***Abstract*—The architecture described in this paper provides emotional, believable agents for a serious game environment. Agents are designed to help players learn skills and to evaluate performance. The framework has been designed with a close connection to the development of other system components: the Game Engine for creating the virtual world; the Narrative Engine for driving the story; and the Dialogue System for communication. In order to provide a synchronised, intelligent solution it is essential to adapt traditional designs for independent components to consider the system as a whole. Effective communication and synchronisation between system components is created by introducing a new system element: the Translation Engine. In this paper we describe the Agent Framework and associated services within the Translation Engine. This includes communication to other relevant system components but the components themselves are treated as *black boxes*.**

I. Introduction

Staff training using tailored competence development is recognized by many organizations as a key business success strategy. Current approaches that minimise the *time-to-competence* are resource intensive in terms of both development and delivery. The main aim of this project is to develop a new genre of Technology Enhanced Learning environments that will support rapid competence development of individuals within the domains of innovation and project management. For example The Universal Competency Framework [1] lists competencies for project managers to include *Leading and Deciding*, *Organising and conceptualising*, and *Enterprising and Performing*. This work presented in this paper is part of the research carried out in the Transformative, Adaptive, Responsive and enGaging EnvironmenT (TARGET) [2] European project. The TARGET platform consists of a set of innovative tools and services. The learner is presented with complex situations in the form of game scenarios. Interacting with the scenarios results in enriched experiences that lead to the development of competencies.

Within the context of a serious game for this novel Virtual Business Environment (VBE) we present a FIPA compliant [3] agent framework. The framework primarily supports two types of agents. The first are cognitive-intelligent and social-emotional Non Player Characters (NPCs) within the game, for example co-workers. The second are Artificial Mentors (AMs) that help guide the player through a changing and challenging learning environment, for example as a support to a real (human) mentor. NPCs interact with players and contribute to driving the story, thus they can be used as a means of managing the flow of the narrative. AMs are a tool that guides players and analyses progress. In such a way, agents are a fundamental aspect of the learning experience not simply a means to enhance the interaction for users.

The serious game environment described in this research has five main components: an Agent Framework for the characters; a Narrative Engine for the story; a Game Engine for the visualisation and simulation of a virtual world; a Dialogue System to enable communication between characters (humans and NPCs); and a Translation Engine that is responsible for effective synchronisation and communication.

A Game Engine is responsible for creating the virtual world inside which agents live. Combining a Multi-Agent System (MAS) and Game Engine design in a mutually supportive approach is a relatively new approach in games environments [4]. Designs considering the strengths and weaknesses of both systems are an important step in providing real-time games with cognitive intelligent NPCs. The VBE described in this research extends the work of Dignum et al. [4] to include two other important system components: a Narrative Engine and a Dialogue System.

The Narrative Engine is responsible for supporting engaging stories where the user can experience complex learning situations. A story can be adapted dynamically to reflect the individual player's performance. The storyline is adapted in response to events that are generated either by agents or by the Narrative Engine. NPCs help to drive the story by creating or engineering situations consistent with the story, for example meetings, deadlines, confrontations, collaborations, etc. This is a two-way interaction where situations generated by the Narrative Engine can also influence an agent's behaviour.

Communication between characters, both those controlled by agents and users, is mediated via a Dialogue System. The connection, therefore, between the Agent Framework, the Dialogue System, Narrative and Games Engine is extremely important and a major design consideration. In Figure 1, this connection is identified as a Translation Engine and is composed of a set of modules that provide services for all system components. This novel approach facilitates communication and synchronisation between the Agents, the Narrative and Game Engines, and the Dialogue System whilst allowing each one to be substituted with minimal impact to the system as a whole. Substituting any of these system components leads to a wider set of solutions the TARGET environment can provide. For example, the game may take place in a 3D virtual

978-0-7695-3986-7/10 $26.00 © 2010 IEEE
DOI 10.1109/VS-GAMES.2010.25

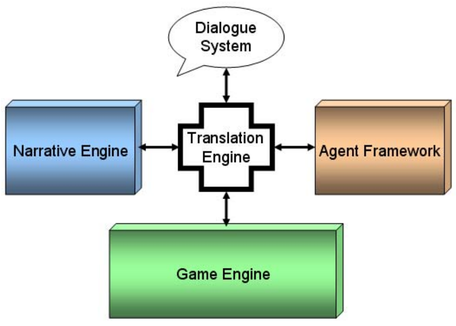

Fig. 1. System Overview

environment more familiar with video games or Second Life, or equally, may have a web based visualisation.

It is important to note that agents are autonomous and are not centrally controlled. The Translation Engine provides a set of services that facilitate interactions between agents and other system components, it does not manage them.

The research presented in this paper focuses on the Agent Framework within this context; specifically describing agents that are developed to create goal directed NPCs and AMs for competence development. The scope includes agent-related components of the Translation Engine because these are agent specific services and are FIPA compliant. A full description of the Dialogue System, Game and Narrative Engines is beyond the scope of this paper and, consequently, they are viewed as black boxes. The Agent Framework, however, has been carefully designed so that it may be integrated into a number of different systems with the prerequisite that support is provided for combining stories and simulation.

This paper is organised as follows: in Section II the previous work is discussed; Section III provides an overview or the system justifying our design decisions; Section IV gives an overview of the Translation Engine in the context of the agent framework, to give a higher level view of how the system components interact; Then, in Section V the agent framework is described; Section VI describes the Blackboard implementation and how it relates to the agent framework and the Translation Engine.

II. Previous Work

Agents are autonomous entities that observe their environment and perform actions to satisfy goals [5]. Groups of agents in a Multi-Agent Systems (MAS), in the context of computer games, are used to represent characters that play roles. In MAS literature a *blackboard* [6], [7] is a shared memory or common area where agents are permitted to both read and write. In a traditional blackboard system [6] a central control unit (*scheduler*) is an integral element.

Agents and MAS are commonly used to model characters in games. Characters in stories or games are often human or are anthropomorphised. Modelling mental attitudes is, therefore, a necessary step in creating believability and player immersion. Mental-attitude modelling is a common approach in general MAS research for problem solving in complex dynamic environments [8]. Agents are given human mental parallels using: *Beliefs* to represent information about their environment; *Desires* to represent motivations; and *Intentions* to represent deliberating and goal achieving. This architecture is called a BDI approach [9]. It is particularly relevant when the environment is a game and the agents represent characters. Agents can appraise and react to situations in a natural way with an element of variability [10]. Although BDI agents are very powerful they do not include memory and emotion in the decision making process.

Several avenues of research extend BDI agents to include an emotional component. Pereira et al. [11] present an extension called *Emotional-BDI* Agents that includes an internal representation of capabilities and resources to make agents self aware. Capabilities are abstract plans the agent can act upon; resources turn the abstract plans into actions the agent can perform. Jiang et al [12], [13] present *EBDI* (also Emotional BDI) agents by adding primary and secondary emotions into the decision making process of a traditional BDI approach. Primary emotions are used as a first filter to adjust the priority of beliefs, allowing agents to speed up decision making. And secondary emotions refine the decision when time permits.

One of the most common emotion modelling approaches was developed by Ortony, Clore and Collins [14] and is known as the OCC Theory of Emotions. It is a cognitive computational model for eliciting emotions that specifies 22 distinct emotion types or categories. The OCC theory suggests that the emotions a person experiences depends on what s/he focuses on in a situation. Emotions are *valenced* (positive or negative) *reactions to events, agents or objects*; the strength of an emotion is dependent on any stimuli in the environment. The OCC model has been established as the standard model for synthesising emotions in (agents) characters.

The original version of the theory is often criticised as being both overly complex and also lacking other important features [15], [16]. Ortony, in a follow up publication [15] suggested consolidating some of the 22 original categories into ten specializations: five positive and five negative. He suggests that these categorizations are sufficient to create believable characters and personalities in games and avoids many of the complexities of the previous method, which can sometime lead to characters being unconvincing. The combined positive categories concern good situations:

- about the possibility of something good happening (hope)
- because a feared bad thing didnt happen (relief)
- about a self-initiated praiseworthy act (pride, gratification)
- about an other-initiated praiseworthy act (gratitude, admiration)
- because one finds someone/thing appealing or attractive (love, like, etc.)

The negative categories concern bad situations:

- about the possibility of something bad happening (fear, etc.)

- because a hoped-for good thing didnt happen (disappointment)
- about a self-initiated blameworthy act (remorse, self-anger, shame, etc.)
- about an other-initiated blameworthy act (anger, reproach, etc.)
- because one finds someone/thing unappealing or unattractive (hate, dislike, etc.)

Ortony [15] suggests that these categories have enough generative capacity to embody believable and realistic emotions for affective agents. He also mentions that a key issue is for agents to parse their environment and understand important aspects of it that will elicit emotions. Ortony repeatedly stresses that in order for agents to be believable they must be consistent. Bartneck et al. [16] in their critique of the original OCC Theory mention that an important omission necessary for consistency is a history. The emotion model needs to keep a history of events, actions and objects. The history can be used, amongst other things, for calculating the perceived probability of a future event. The history can also be used to monitor a character's performance as it attempts to achieve its goals and also to calculate effort.

There are a number of other agent architectures that base their model of emotions on the OCC model. FAtiMA (FearNot! Affective Mind Architecture) [17] incorporates a reactive component for expressing emotions and influencing an agent's decisions. Parunak et al. [18] extend the BDI agent framework with the essential features of the OCC model to create the Disposition, Emotion, Trigger and Tendency (DETT) model. The DETT model is for situated agents and extends previous models to include disposition to distinguish an agent's susceptibility to various emotions. An extension of FAtiMA is ORIENT (Overcoming Refugee Integration with Empathic Novel Technology) [19] which balances physiological and cognitive dimensions for creating believable agents that show empathy.

An often overlooked but extremely important aspect of agent design within a complex virtual environment or a serious game is the relationship between the game engine and the agent architecture. Dignum et al. [4] discuss the relationship between two essential system components with inherent integration difficulties: game engine and agent framework. Games engines have concerns such as real-time display of complex graphics in complex environments, e.g. involving many objects or elements, collision detection, physics, etc. MultiAgent platforms generally are concerned with autonomy of intelligent agents that have time to reason and deliberate on a suitable course of action. Dignum et al. stress the importance on designing the two components so as to anticipate problem areas such as: synchronisation, information representation and communication.

The agents in this framework are developed to be FIPA compliant. FIPA (Foundation for Intelligent Physical Agents) [3] is an IEEE organisation for agents and multi-agent system standards. FIPA maintains a collection of standards which promote interoperation of heterogeneous agents and their services.

III. System Design

The architecture described in this research is principally designed to address the issues of synchronisation, information representation and communication as identified by Dignum et al. [4]. We extend the work of Dignum et al. to include other essential system components in this novel Technology Enhanced Learing environment: the Narrative Engine, the Dialogue System and the Translation Engine.

The main focus of this paper are the agent framework and the Translation Engine. There are two main types of agents: NPCs and AMs. Their view of the environment (or virtual world) is provided through the translator, and is made up of information relating to communication with both the narrative (the story) and the Game Engines (graphics pipeline, etc).

The VBE supports business entities similar to real organizations and characters similar to real people. Players, therefore, expect NPCs to behave in a realistic manner that is consistent with a personality and a role. A character's behaviour should persist throughout the game, not just while visible to the player. This is particularly important in our VBE as NPCs play characters with human-like personalities and roles that are crucial to the story development. Also, there can be a number of NPCs acting in a game and each one should have their own identifiable, consistent, believable, autonomous actions and behaviours.

Maintaining a set of characters of such complexity consumes many system-resources. Therefore, it is not appropriate to combine with the scarce system resources of a Game Engine [4]. The Game Engine controls elements such as the rendering pipeline, collision detection, avatar representations, etc, and system cycles are organised according to these considerations. Agents require computational freedom to reason about their goals and adapt plans to a changing environment. This is not conducive to inclusion in a game design where agents are required to make decisions in one system time step or cycle. For this reason an asynchronous solution is more appropriate, where agents run in a separate thread from the Game Engine. Following on from this design decision, a similar approach is applied to the Translation Engine. It has the potential to be a bottleneck in the system if not implemented correctly. For this reason it is decomposed into smaller independent modules that operate in a multithreaded environment.

The Narrative Engine is also modelled separately from the agents and the Game Engine. Agents can produce events to drive the story in the Narrative Engine (and vice versa), but it also has a number of other concerns. Logically, therefore, the agent framework and Narrative Engine are separate components. Likewise for the Dialogue System, different game scenarios may have different dialogue requirements, and these should not be tied into the reasoning engine of the agents.

There are a number of areas that require close attention when describing an agent framework that operates in such an environment. The game is dependent on the successful

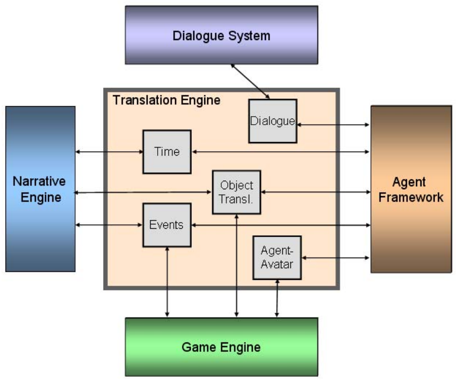

Fig. 2. A Subset of the Modules inside the Translation Engine, from the point of view of the Agent Framework.

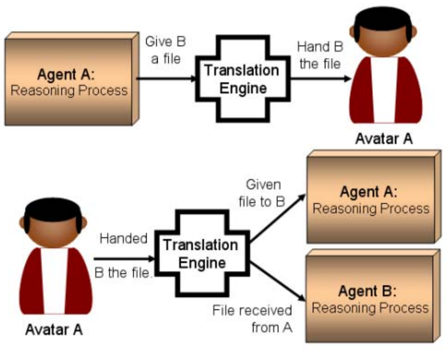

Fig. 3. Mapping Agent to Avatar. NPC Adam has a Reasoning Process A and an Avatar A. *Top row*: Adam's reasoning process sends a message to pass Bob a file. *Bottom row*: After handing over the file, Adam's reasoning process is updated with the handover and Bob's with the receipt of the file.

interplay of each of these independent and interdependent elements. It is therefore very important to define communication channels and ensure that the Translation Engine carries out the appropriate tasks in a timely manner.

IV. Overview of Translation Engine

The Translation Engine is composed of a set of modules that perform dedicated tasks and services. This section describes the subset of modules that are related to the Agent Framework and its integration in the VBE. These modules are described as performing services for Agents.

The Translation Engine is responsible for maintaining consistency between the Agent Framework, the Narrative Engine, the Game Engine and the Dialogue System. It creates mappings from: agent to avatar; game objects to agent object; Dialogue System to agent communication; and also translation of time between game, agents and narrative.

For example, if a player walks into an important meeting with an NPC there are a number of places where consistency must be maintained: the Game Engine must display that the avatars are *physically* in the same room, where they sit, etc; the Narrative Engine is sent an event to identify that an important meeting has taken place and therefore the story is advanced; the Dialogue System is used for greetings and conversation-type communications between the characters; and the agent framework must consider actions to perform that are consistent with the NPC's personality and role.

The Translation Engine must accept the input from each of the system components and translate them to appropriate representations for delivery to the relevant recipient(s), see Figure 2. Note that relationships are not simple one-to-one mappings between components and often include many messages to many components. In essence, the translator is responsible for message passing, although its purpose is complicated by the requirement of each of the system components for different abstractions of the data. Each component, therefore, has a slightly different abstraction, or view, of the environment.

In particular, an important aspect of an agent's interaction with the Narrative Engine (to drive the story) is the notion of events. The translator must connect the representation of events as understood by the Narrative Engine, the agents and, where relevant, the Game Engine. In the above example an event could be the initiation of the meeting; or the sharing of a document by the two characters.

The modular approach to the Translation Engine allows the replacement of any of the other system components with minimal impact. This is very important to the TARGET environment because different organisations have specific requirements. Dialogue System options are currently either simple multiple choice or a natural language system. The Game Engine choices are currently either a stand alone or web based application.

A. Mapping Agent to Avatar

Each NPC has a reasoning process within the agent framework and also an avatar in the Game Engine. The avatar is little more than a visual representation of the agent; it acts in the VBE and is also given simple reactive behaviours such as unconscious Non-Verbal Communication (NVC) like blinking and breathing. The agent framework contains the reactive and deliberative behaviours of the character. The agent's goal oriented reasoning engine is described in Section V. It is extremely important that the translator ensures that the agent reasoning and avatar behaviour are synchronised.

Inter-agent communication, therefore, does not take place inside the agent framework. The translator ensures that any avatar behaviour in the VBE is interpreted and updates are sent to both its own agent reasoning component and that of any other relevant agents. In Figure 3 the character Adam has a Reasoning Process A and an Avatar A; Adam wants to give Bob (B) a file. Figure 3 *top*: Adam's reasoning process sends a message to the avatar to pass Bob the file. Figure 3 *bottom*: after handing the file to Bob an update is sent to Adam's reasoning process and also to Bob's to acknowledge receipt. Likewise if a response is generated from the agent reasoning process, the translator propagates this back to the agent's avatar and also to any other relevant agents or avatars.

B. Mapping Object Representations

The translator is responsible for the representation of objects from the game environment to NPCs in the agent framework. In other words, NPCs must be able to react to objects that exist within the Game Engine. A character, for example, may pick up a file placed on a desk. The translator module is also responsible for mapping the representation of objects from the game environment to the agent framework.

If the shared object from the above example is a spreadsheet it may be represented visually as a paper file, CD, or memory stick, for example. The agent does not need to store the same information about the spreadsheet but rather identifies that a particular character has shared a specific important document, for example. The act of sharing may also trigger an event in the Narrative Engine.

C. Mapping Dialogue

There is not a direct correlation between what the agent understands as dialogue and what the dialogue system will provide. The Translation Engine also contains a module to create a mapping between these two elements. In the VBE novice and more experienced players can use different versions of the Dialogue System, allowing different degrees of control over the sentiments or words used. The dialogue itself is not necessarily important for representation in the agent reasoning, rather the result of the dialogue. For example, if a conversation or dialogue exchange creates an atmosphere of collaboration, camaraderie, mistrust or displeasure this is reflected in the NPCs emotional state and opinion.

D. Mapping Time

The discussion on time has two elements: maintaining consistent time across different system components and also ensuring timeliness in interaction. It is not related to system cycles. The Agent Framework and the Narrative Engine are the components that require coordination, in terms of driving the story through NPC interaction with players.

Time in the game world is not necessarily consistent with the real world. Also time in the game is variable, i.e. non-linear. For example, when having a conversation with another character real time can be used, but a player does not have to stop playing because it is night time in the game; the player can fast forward until the next morning, a project deadline or indeed to any appropriate point in the game.

Another aspect to time management is to ensure that the reaction time of agents is within a reasonable margin. Game and agent system cycles are not connected, but characters in the game expect a somewhat timely response from NPCs. The translator is responsible for scheduling events and identifying when inappropriate wait times are experienced; for example where a character takes too long to answer a question or share an important document.

E. Events

Stories are emergent not fully scripted or based on a branching structure. They give the player a setting and a goal, but without strict instructions or set of tasks. The story is driven by situations like meeting other characters, performing jobs or satisfying deadlines. Events are therefore responsible for driving the story. For example such events can be deadlines set up by the story creator, or equally interactions with NPCs, such as a conversation or meeting. The translator is responsible for creating the link between these differing representations because each of the agents, the Narrative Engine and the Game Engine can all produce events.

V. Agent Framework

Agents represent autonomous NPCs and AMs in the game. They represent believable characters that interact with the player and drive the story. They must have goals and motivations and have behaviour that is consistent with these. Their believability is dependent on their ability to display a variety of human emotions and maintain particular personality traits.

Agents in this research are FIPA compliant and represent autonomous, communicating software elements that use a common language and have a set of services to support them. Many of the agent services are performed by the Translation Engine, particularly communication, as described in Section IV.

In this section we describe the agents and their emotional basis along with the communication strategy from the point of view of the agent reasoning engine.

A. Agent Roles

The two main types of agents supported by the framework are NPCs and AMs. NPCs have an embodiment within the VBE interacting with the environment and the other game entities. The *brain* and *body* of the agent is split between the Agent Framework and the Game Engine (avatar), respectively. NPCs play roles of human personalities in story setting. Consequently, these agents are not only cognitively intelligent, but are also socially and emotionally aware.

Characters do not necessarily take the form of a lecturer or a teacher in the traditional sense. They can, in specific circumstances, be used to give a tutorial on certain subjects. Their most important use, however, is to play a part in a situation similar to role playing scenarios that many organisations currently employ. NPCs interact with the player as another character to teach skills through experience. Characters can be work-mates, competitors, customers, subordinates, superiors, or any other relevant human role. It is important that characters remain consistent in their representation of a role to maintain the player's believability. A complete team can also be modelled to teach the player about working in a team; organisational hierarchies can equally be applied. The size of the set of NPCs is defined by the designer of the game scenario and is not limited to individual teachers or tutors. It is through these types of varied interactions that players are exposed to circumstances where they need to learn a particular skill or competency. These interactions also contribute to the driving of the narrative. The game scenario designers identify the important narrative-driving elements related to agents or personalities and associate events with them.

In organisations learners often have a mentor to help in the learning process and to identify competencies they should have. The AM in this framework does not aim to replace this person, rather to augment the mentor-learner relationship. The AM is responsible for supporting the mentoring process of the player's learning environment and affecting the VBE with the aim of assisting the user to achieve their learning aims or competencies. Associated to each user is one AM that monitors the performance of the individual. Monitoring the player allows the AM to identify points during play where game settings should be altered, for example the player may be exhibiting signs of frustration and requires a hint or some type of help. The data collected by monitoring the game play is used by the AM to evaluate the learning objectives and report results to the player and the human mentor.

B. *BDI Agents*

We use BDI agents because they are goal-oriented and are good at establishing a balance between deliberative and reactive behaviours; formulating a plan and executing it. Agents formulate plans to satisfy goals they wish to achieve. Satisfying a goal involves decomposing it into a set of sub-goals, which in turn are broken down into one or more actions. Actions available to agents are consistent with the personality and role encoded in their internal data, using beliefs and desires. Performing these actions or intentions, therefore, takes the agents closer to achieving their goal.

BDI agents do not inherently have provision for expressing emotions. The BDI framework is extended to include an emotional component that is evaluated in a similar way to deliberating as described by *Intention*. In Figure 4 the Environment, Beliefs, Desires, Intentions and Emotions are shown as states. Sensing and Perception, Appraisal, Reasoning Engine and Actions are processes. An agent senses its environment and perceives certain aspects of it. In some situations an impulsive action may be performed. Otherwise the perception information is used to update the agent's view of its environment, or its beliefs regarding its surroundings. An appraisal of these beliefs with regards to emotional states can lead to a re-evaluation of the environment and an updating of the agent's beliefs. The reasoning engine examines information about the agent's beliefs, emotions and desires about where it would like to be and under what circumstances. This can again lead to a re-appraisal of an agent's beliefs and therefore its emotional state. Upon examining all of the agent's internal data in the form or beliefs, desires and emotions, the reasoning engine will identify a goal and formulate a plan to achieve it. This leads to the agent's intentions being updated. Ultimately a set of actions are performed that have an effect on the environment, which the agent can perceive.

An agent's reasoning *process* can be separated from its expression of emotions. Emotions are considered as a set of states the agent can use in the reasoning process to decide upon actions to perform. An agent's behaviour is modelled by the set

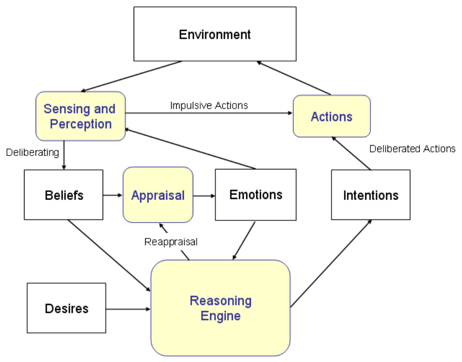

Fig. 4. The agent architecture. Black boxes are states and blue rounded boxes are processes.

of actions it can perform. An agent's personality, therefore, is composed of input from the agent's beliefs, desires, emotions and intentions as well as the set of actions it has available to it, which are selected using the reasoning engine. An agent will have its own memory and history included locally in this structure. Relevant contents of it will be used to update the Blackboard at appropriate times.

C. *Emotional Agents*

Since many of the targeted competencies are based on emotional intelligence and soft skills it is important to have believable synthetic characters in an engaging learning environment. Believable characters must be consistent and coherent in their expression of behaviours, especially where they are related to emotions.

The OCC [14] Theory of Emotions is commonly used to represent emotions in agent frameworks. The theory as presented in the original publication is not completely appropriate for agents in this project. There are a number of areas where the theory is overly complex for our requirements. The adaptations presented by Ortony in [15] are more appropriate for our research. Specifically the ten specializations of feelings Ortony consolidates from the original theory (see Section II). There are five positive and five negative reactions based around something either good or bad happening.

Our method of analysing these emotions to create reactions also uses the Ortony model. Emotion response tendencies are broken down into: expressive, information processing; and coping. Expressive emotions deal with postures and gestures and also both verbal and NVC. Information processing deals with distraction or the diversion of attention to the emotion-inducing event. The coping strategies deal with problem solving to bring the situation under control and also with emotional control, concerning either the agent's own emotions or another's.

There are a number of aspects not considered in the original OCC theory that are important to our agents. As identified by Bartneck [16] an important omission from the OCC model is a history of events, actions and emotions. As well as using the history for calculating the likelihood of events happening and desirability of events actions or objects the history can be used for a great deal more. The history is a very important part of our agent framework in the wider context of the AM for analysing the player's performance. The history will also be used to maintain agent consistency in behaviour and personality, as that is one of the most important elements to maintaining believability of agents.

Emotions communicate information about characters and also can be used to create new goals e.g. in conflict resolution situations a goal may be to calm down or when learning management skills the goal may be to encourage people.

D. Communication

Traditional agent communication is carried out using a blackboard. Agents can communicate a wealth of information to one another or to other parts of the system. A message is a structured piece of information sent over an agreed communication channel. A message can contain information such as agent location, instructions, sharing data about the environment, etc.

As we have discussed in Section IV, agent communication is carried out using the Translation Engine. This ensures that all system modules maintain a synchronised view of the environment and representation of each other. Therefore traditional message passing techniques from agent literature are not relevant in this context. The most appropriate form of communication and information sharing is the BlackBoard.

The framework described in this paper includes a blackboard for sharing relevant information and also for storing vital data about agent history and each player's performance. The blackboard is situated within the Translation Engine, see Section VI, and is bound by the rules that make it FIPA compliant.

The communication model must ensure agent believability when playing the game. As agent-agent interaction can be visible to the player, communication methods include those more familiar to human-agent situations. This will mean using the same Dialogue System as players, for example the dialogue between two NPCs in a meeting should be available to the player.

Agents also need to communicate with the Narrative Engine. The method of communicating is in the form of events. Events can identify a number of elements that are important to advancing the story; such as scheduled meetings, deadlines, and attitudes or relationship descriptors e.g. antagonism, cooperation. The translator will provide the services for communicating with other game components.

Finally agents also communicate with the Game Engine. The environment, in terms of the game setting (virtual world), is represented in a physical manner by the Game Engine. Contrast this with a more abstract representation that the agents are aware of.

VI. Blackboard

A blackboard is an application that contains a common data repository that is iteratively updated by agents. In general Blackboards have three main components: the blackboard repository; agents or knowledge sources that populate the repository; and a control shell that moderates the activity regarding the repository and the data it contains. In the research presented in this paper, the agents act as knowledge sources and the Translation Engine provides the location for the blackboard and also the control shell.

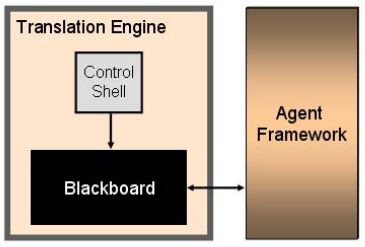

Fig. 5. The Translation Engine contains the Blackboard and the Control Shell.

The Blackboard presented in this research is used for communication and also to store player's performance data. Agents can add, update and erase data from the common areas. Agents do not communicate directly with one another, all information is shared via the blackboard. By default information on the blackboard is public, but private data or messages can be shared by agents upon request.

Agents update data in the blackboard and publish a notice about the update. Interested agents can subscribe to these messages to be notified of changes to the blackboard. This method is adopted to provide inter-agent communication. The publish-subscribe mechanism is automatically initiated for agents in a game where play depends on communication between characters.

The Blackboard also contains some information about a player's performance and the AM uses it under several circumstances: in-game automatic recommendations; specific user request for assistance; and final analysis of player's performance. The AM uses this data to make recommendations and for evaluation. The AM can recommend changes to the gameplay while the game is being played to react to user's performance, for example if the player appears frustrated. The Blackboard data is also referenced when a player calls for assistance during gameplay. Finally, the data is used when the player or a human mentor wish to analyse the player's performance to evaluate whether the player has learned the necessary skills.

The Translation Engine handles communication between game elements. The Blackboard and its control shell are therefore implemented as a module of the Translation Engine. It is a separate module so as to minimize any potential performance bottleneck issues related to both the Blackboard specifically and the Translation Engine as a whole. A further benefit of designing the architecture this way is related to the different levels of abstraction required by the agent framework and other system components such as the Game Engine.

Upon completion of a game scenario, or chapter, the control shell determines which of the Blackboard data is relevant to be updated in a more permanent repository, for example in the player's history.

VII. Conclusions

In this paper we have presented an agent framework in the context of a modular environment for a serious game. The major concerns with this architecture are to ensure, primarily, that design of the Game Engine and agent framework are developed in synchrony with one another to ensure seamless interoperability of each of the component parts. As the final game environment also includes a Narrative Engine with emergent story properties, consideration has been given to the required lines of communication as regards story driving events. We have introduced a Translation Engine that is composed of a set of individual modules performing the larger task of system synchrony and interoperability.

The Translation Engine will be implemented as a collection of modules that run in different threads to avoid, as much as possible, being a bottleneck for the system. Conceptually these modules make up the Translation Engine, but practically they are separate system components.

A main motivation of the system is to provide believable, emotional agents. For this reason we are using an extension to the proven technique of BDI based agents with a modified OCC Theory of Emotions.

The future work of the TARGET project will define the building blocks of the novel Narrative Engine and the new approach to the Dialogue System. These system components will be integrated into the system in a similar manner to the modular approach described in this paper for the agent framework and the Translation Engine.

Acknowledgments

The authors thank the respective project partners, as well as the European Commission for all scientific, organisational and financial support. The TARGET project is partially funded under contract number FP7-231717 within the Seventh Framework Program.

References

[1] SHL, "Universal competency framework," (2010, Jan). [Online]. Available: http://www.shl.com/WhatWeDo/Competency/default.aspx

[2] "Project objectives," (2009, Oct). [Online]. Available: http://www.reachyourtarget.org

[3] "Standard status specifications," (2002). [Online]. Available: http://www.fipa.org/index.html

[4] F. Dignum, J. Westra, W. A. van Doesburg, and M. Harbers, "Games and agents: Designing intelligent gameplay," in *International Journal of Computer Games Technology*, 2009.

[5] M. Wooldridge, *An Introduction to MultiAgent Systems*. Chichester, West Sussex, England: John Wiley & Sons, Ltd, 2002, iSBN: 047149691X.

[6] B. Hayes-Roth, "A blackboard architecture for control." *Artificial Intelligence*, vol. 26, pp. 251–321, 1985.

[7] I. Craig, "A new interpretation of the blackboard architecture, technical report 254 dept computer science, university of warwick," 1993.

[8] A. S. Rao and M. P. Georgeff, "Bdi-agents: From theory to practice," in *First International Conference on Multiagent Systems (ICMAS'95*, 1995.

[9] M. Bratman, *Intentions, Plans, and Practical Reason*. Harvard University Press, 1987.

[10] P. N. N.P. Davies, Dr Q.H.Mehdi, "Creating and visualising an intelligent npc using game engines and ai tools," in *European Conference on Modelling and Simulation (ESM05)*, 2005.

[11] D. Pereira, E. Oliveira, N. Moreira, and L. Sarmento, "Towards an architecture for emotional bdi agents," in *Artificial intelligence, 2005. epia 2005. portuguese conference on*, 2005, pp. 40–46.

[12] H. Jiang, J. M. Vidal, and M. N. Huhns, "Ebdi: an architecture for emotional agents." in *International Conference on Autonomous Agents and MultiAgent Systems*, 2007.

[13] H. Jiang, *From Rational to Emotional Agents: A Way to Design Emotional Agents*. Saarbrücken, Germany, Germany: VDM Verlag, 2008.

[14] A. Ortony, G. L. Clore, and A. Collins, *The cognitive structure of emotions*. Cambridge University Press, 1988.

[15] A. Ortony, *On Making Believable Emotional Agents Believable*. MIT Press, 2003.

[16] C. Bartneck, M. J. Lyons, and M. Saerbeck, "The relationship between emotion models and artificial intelligence," in *SAB2008 Workshop on The Role of Emotion in Adaptive Behavior and Cognitive Robotics*, 2008.

[17] J. Dias and A. Paiva, "Feeling and reasoning: A computational model for emotional agents," in *Lecture Notes in Artificial Intelligence*, vol. 3808, 2005, pp. 127–140.

[18] H. Van Dyke Parunak, R. Bisson, S. Brueckner, R. Matthews, and J. Sauter, "A model of emotions for situated agents," in *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 2006, pp. 993–995.

[19] M. Y. Lim, J. ao Dias, R. Aylett, and A. Paiva, "Improving adaptiveness in autonomous characters," in *IVA '08: Proceedings of the 8th international conference on Intelligent Virtual Agents*, 2008, pp. 348–355.