To Date or Not to Date? A Minimalist Affect-Modulated Control Architecture for Dating Virtual Characters

Michal Bída, Cyril Brom, and Markéta Popelová

Charles University in Prague, Faculty of Mathematics and Physics,
Department of Software and Computer Science Education
Malostranské nám. 2/25, Prague, Czech Republic
michal.bida@gmail.com, brom@ksvi.mff.cuni.cz,
PopelovaM@seznam.cz

Abstract. As part of our broader initiative on promoting the education in the field of IVA control mechanisms at high schools and universities, we have created a micro-game Cinema Date, which introduces students challenges posed by controlling 3D virtual characters and expressing their emotional state. The game features two virtual teenagers dating on their way to the cinema. The player can influence the course of the date by influencing behavior of the boy. Existing IVA architectures did not satisfy our requirements on the architecture being reasonably simple, yet capturing affect-modulated behavior, transition behavior and future-directed intentions. Here, we present the game, focusing on the minimalist control architecture of its main characters.

1 Introduction

With the field of intelligent virtual agents (IVAs) maturing, a limited number of tools supporting education of students entering the field is becoming increasingly problematic. To our knowledge, out of many agent-authoring tools only Storytelling ALICE [1] and NetLogo [2] address the issue of education explicitly. However, Alice is oriented on teaching primary and middle school children the programming basics and uses 3D virtual reality as a means rather than an educational object, and NetLogo, while being an excellent entry-level tool for building simple agents and running social simulations, is not well suited for building 3D agents with complex behavior.

To fill the gap, we adopted a long-lasting aim to develop educational applications suitable for advanced high-school and university students for improving their skills in programming high-level behavior of 3D IVAs. Our main project, Pogamut [3], a tool enabling a rapid development of IVAs based in worlds of first person shooter action games, has been already adopted as an educational platform at several universities. Despite generally positive comments [4], Pogamut has a limitation: it is oriented on action game AI. Thus, under the umbrella name Emohawk, we are now coming with a new set of tools featuring a less violent content and addressing more issues connected with development of IVAs, including, e.g., emotion modeling. Two such tools have been already finished and released: StoryFactory [5], an application supporting teaching high-school and non-programming university students (e.g., new media art)

H. Högni Vilhjálmsson et al. (Eds.): IVA 2011, LNAI 6895, pp. 419–425, 2011.

basics of 3D animations by developing machinimas, and SteeringTool [6], a simulation for teaching the topic of IVA navigation. Meanwhile, as a prequel to the Emohawk package, we have developed a micro-game Cinema Date to draw attention of students to several key IVA issues. The focus of this paper is this game.

In the present version of the game, a simple narrative is played out by two IVAs: Barbara and Thomas. The player observes the narrative from the third-person perspective (Fig. 1). At the beginning of the story, the characters agree to go to the cinema, and during their approximately two minutes long walk there, the player has the opportunity to influence the behavior of the boy towards the girl. In case of no player input, the characters will be engaged in a casual dating conversation; however, the player can make the characters to argue with each other by making Thomas acting strangely. The story has two possible ends: either the characters make it to the cinema together or Barbara breaks up with Thomas, which is the player's goal. The exact course of the story is emergent and depends on the player's actions, the characters' current state and a limited random element. Note that the story will actually have four variants in the final version of the game: a player will be allowed to choose on behalf of which character to play and whether the game goal is negative or positive (i.e., to reconcile an initial tension between the characters).

The IVAs act in a virtual city we developed for UnrealEngine2Runtime. For affect simulation, we use the ALMA model [7] and recognize about 50 events triggering eight OCC [8] emotions. Characters exhibit eight different complex behaviors that are triggered by about 20 reactive rules. The behaviors are expressed by means of about 200 mo-caped animations, 50 emoticons and colored bubbles around characters heads expressing the characters' overall feeling. Examples of actions include: joke, compliment, insult, slap, apology, question, speaking, laugh, kiss, touch, bump, etc. The player can make Thomas a) to perform a positive or a negative action to Barbara, b) to increase or decrease his distance from Barbara, c) to change the angle in which he is following her, and d) to switch between a normal walking and a "silly" walking style. Barbara's reaction depends on her emotional state and the action of Thomas, e.g., when Thomas starts walking silly, Barbara may ask him to stop it. Her action may also trigger a reaction of Thomas, resulting in a short sequence of actions between the characters (with the player triggering the chain with the first action).

When specifying this scenario, we had several goals in mind. We wanted to show students that IVAs are fun and life-like, to immerse them in a VR environment and to motivate them to play with the scenario and explore its possibilities. Though the game is short, its state space is already large. From the pedagogical standpoint, we wanted to highlight the distinction between an *autonomous* agent and a user controlled 3D VR puppet (with which students become familiarized using StoryFactory tool). Additionally, the game, when supplemented with a teacher's explanation, introduces students the issues of IVA navigation, emotion modeling and reactive behavior.

Our major technical issue was the development of the IVAs' control architecture, balancing its complexity so that we can describe intended behavior but not burden the designer during development with the architecture's superfluous features or wasting computational resources in run-time due to the architecture's superfluity. Additionally, the architecture should serve well for demonstrative purposes regarding novices to the IVA field. Different architectures suit different purposes, as highlighted by the empirical study [9].

We required the architecture 1) to allow us to define the overall story shape yet to generate behavior in an emergent manner within the story boundaries. We further needed to handle: 2) reactive behavior with transitions (to swiftly change behavior and depict a transition behavior), 3) affective behavior (to portray emotions), 4) occasional future-directed intentions (to make the overall behavior more persistent), 5) a limited user interaction, 6) synchronizing the characters. Solutions friendly from the designer's perspective and operating in a timely fashion, such as finite-state machines (FSMs), behavior trees [10] or the reactive planner POSH [11], are insufficient due to Requirement (4) and partly (2), (3) and (6). Advantages of complex solutions, e.g. [12, 13, 14], addressing issues beyond our needs, such as equipping agents with general planning abilities and/or making them plausible emotionally and cognitively, comes at a price: increased design time and/or slower real-time computation. Complex reactive approaches that work in a timely fashion, such as ABL language [15], can still overburden the designer. Additionally, these solutions may be too complex for entry-level demonstration.

Thus, we have developed our own control architecture for IVAs: an affect-modulated action selection mechanism working with transition behaviors, future-directed intentions, and with a very simple "drama manager" for synchronizing the characters and making high-level adjustments to the story in real-time. Technically, our mechanism can be conceived as an extension to classical finite state-machines and simple rule-based systems. Its strength lies in adding several features without which would the development of stories of the Cinema Date's complexity be problematic.

The goal of the rest of this paper is to present this architecture. It is detailed in Section 2. Section 3 discusses the architecture's strengths, limitations, and scalability.

Fig. 1. Cinema Date examples. Upper left: an overview of the city. Upper right: Thomas performs "silly" walking. Lower left: Characters argue. Lower right: Thomas is angry.

2 Control Architecture of Cinema Date's Characters

The architecture features 2 kinds of procedural entities, actions and behaviors, and 3 control modules: reactive factories making top-level decisions, an appraisal module appraising events, and a user interaction module handling the user input (for Thomas).

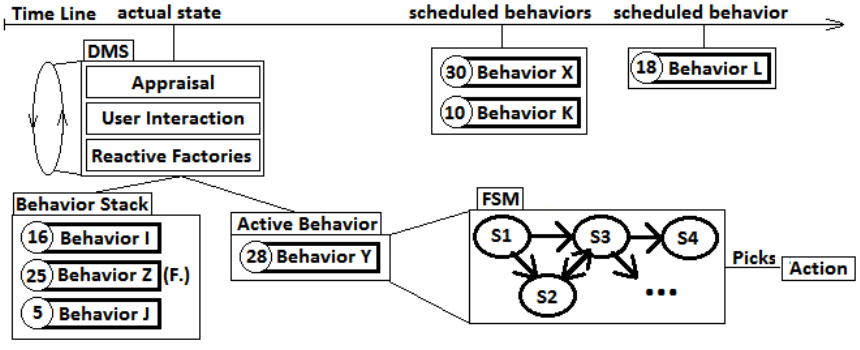

Fig. 2. Control architecture of a single agent. Priorities are given in circles. (F.) represents a frozen behavior. The drama manager is not portrayed here. See text for further explanation.

Actions. Actions are used to capture atomic behavior. Everything our IVAs can do is represented by an action. An example is slapping the other character or laughing (in these cases, the respective action runs an animation and shows an emoticon). Every action goes through an initialization, an execution and a clean up phase.

Behaviors. A behavior organizes actions to sequences to achieve the behavior's goal, which can be, for instance, "to lead the other agent to a certain place". Every behavior has a fixed priority and can be *succeeded* or *failed* with respect to its goal. An agent can have only one behavior *active* at a time.

So far, the notion of behavior is similar to how behaviors are used in other simple control architectures. However, to fulfill Requirement (2) on transitions, we augmented behaviors so that every behavior can pass through the following stages during its lifetime:

- *Init.* This stage executes preparations for the behavior if necessary.
- *Execution.* This is the main state of the behavior life cycle and it executes the normal course of the behavior.
- *Freezing.* It may happen that a behavior with a higher priority (HPB) should take control over the agent and the currently active behavior (CAB) should be interrupted. Before that happens, the CAB's freezing phase is executed, which allows us to specify the initial part of the behavioral transition if needed.
- *Resuming.* If frozen before, this stage is executed first after de-frosting.
- *Finishing.* When the behavior succeeds or fails, clean up actions or a transition to the next behavior can be executed here before the behavior is discarded.

A transition behavior can be executed when a) a CAB is interrupted by a HPB, b) a CAB ends and a frozen behavior is resumed, or c) a CAB ends and a new behavior is

initialized. In each case, the transition behavior has an outgoing and an incoming part, which can be implemented in respective stages of the two behaviors. The two parts can be linked smoothly since the two behaviors are informed about each other.

In order to represent decision making, a FSM is embedded in each stage of every behavior. FSMs in execution stages are complex ones, other FSMs are usually simple.

Behaviors competing for execution at a particular moment are represented on the behavioral stack. Behaviors scheduled for execution in future, i.e., future-directed intentions, are linked with the time-line (Fig. 2).

Decision Making System (DMS). The DMS works in a cycle. Every cycle, three control modules evaluate the events in the environment (Fig. 2). The appraisal module (AM) matches events using reactive rules, appraises them by OCC variables, sends them to the ALMA model as an input and processes the ALMA output emotions. Reactive factories module (RF) use rules to monitor the agent and the environment state and generates new behaviors either on the behavior stack or the time-line, or it removes a behavior from there. The user interaction module (UM) changes (based on the user input) Thomas' parameters, e.g. distance between him and Barbara when he is following her, and generates or removes new behaviors similarly to the UM. After the modules finish their job, the DMS checks the time-line and moves all behaviors scheduled for the current time to the behavior stack (if there are such behaviors). Then, one behavior that will execute a next action is chosen using the following rules (only the first applicable rule is employed):

1. If the CAB has just completed its finishing stage, it is discarded and the behavior with the highest priority is selected as the next CAB from the stack and the execution thread is passed on to it (to its init or resuming stage).
2. If the CAB is in any stage except the execution one, the control is given to it.
3. A behavior with the highest priority is selected from the stack. If it is the CAB, the control is given to it. If it is a different behavior, the CAB's freezing stage starts.

Affect and behavior. The AM processes output ALMA OCC emotions to generate one dimensional value for representing social affect between the two characters. We call this value ranging from -1 to 1 "a feeling". It resembles the pleasure dimension from dimensional theory of emotions, but in our scenario it is valenced to a character. This value is taken into account in individual behaviors and reactive rules in the RF, and it determines the color of the bubbles around the characters' heads.

Representing the story. The architecture offers a designer four key elements for representing a plot (Req. 1). First, the designer starts with capturing the basic story shape by using the time-line for scheduling behaviors with known time of execution, e.g., the designer can set behaviors for Barbara a) to lead Thomas to the cinema at the beginning of the story and b) to call her mother in the middle of the walk. Second, the designer defines a set of reactive factories to monitor the agent or environmental state and generate/remove behaviors accordingly. This mechanism enables two things: executing reactions on some events, e.g., by adding the "kiss girl" behavior after she made a compliment twice and the boy's feeling is high enough, and executing story-important behaviors that do not have a fixed, in advance known time of execution, e.g., "turn right" after the character arrives at a particular crossing. The former may also trigger a short sequence of follow-up behaviors. This mechanism also allows for generating (removing)

future-directed intentions such as if the girl complains that a particular boy's action was silly, the boy will do the same action on purpose half a minute later again. Third, the designer has the same opportunity to add/remove a behavior from within another behavior. Fourth, the architecture features a simple drama manager that allows for synchronizing agents and changing the overall story shape by removing all behaviors from the stack and/or the time-line of both characters at important story points, such as when the couple breaks up (however, we did not use the drama manager extensively).

3 Discussion and Future Work

In this paper we have presented a control architecture for dating characters from a micro-game Cinema Date, a motivational prequel to our larger educational package Emohawk. The architecture is a compromise between simple mechanisms, such as finite-state machines, and complex solutions like ABL language. It goes beyond the simple mechanisms in that it enables easily i) modulating behaviors by emotions, ii) representing transition behaviors, iii) representing future-directed intentions, and iv) synchronizing the characters centrally and adjusting the whole story at important plot points. All of these are important requirements even for short plots featuring several IVAs that express emotions.

Technically, the game served as a case-study project on which we verified that the architecture works well for plots of our complexity. The design time is rather short, though deep testing of the characters' resulting behavior is, of course, needed due to partly emergent nature of the plot. Features (i) – (iii) of the architecture are exploited extensively in the game. The drama manager (iv) has not been employed to its full potential: arguably, it would be needed more urgently in a longer plot with several branches. At the time of writing this paper, we already know the architecture can be scaled well for a similarly long scenario with three characters and more than 15 different behaviors (which we already implemented as an extension to the Cinema Date plot). Scaling it for four characters and longer and branching plots, where the drama manager is expected to be used extensively, is a work in progress. A possible limitation of the architecture for some projects is that it does not feature concurrent behaviors. It also does not employ now popular hierarchical behavioral representation, except the fact that all of our behaviors comprise a FSM (cf. hierarchical FSM and behavior trees). We found the hierarchical approach unnecessary for our purpose.

The game also stands on its own as an educational simulation for quick introduction to the issues of IVA navigation, emotion modeling and reactive behavior. Preliminary evaluation of the game with 5 lecturers/teaching assistants with IVA background suggested that i) the game has indeed a large educational potential as judged by the lecturers subjectively, ii) the lecturers perceived well the internal emotional state of the character, but iii) the game goal can be achieved too easily. Though the easy game-play was intentional, we are currently considering making it more challenging. An evaluation on target subjects is planned, but we first want to have the four variants of the game-play mentioned in Introduction.

The project can be downloaded at: http://amis.mff.cuni.cz/emohawk/.

Acknowledgments. The research related to this application was supported by the Ministry of Education of the Czech Republic (Res. Project MSM0021620838), by a project P103/10/1287 (GACR), by a student grant GA UK No. 0449/2010/

A-INF/MFF, and partially supported by SVV project number 263 314. The name "Emohawk" is inspired by Emohawk: Polymorph II, an episode of Red Dward VI (BBC). The graphical content was created by I. Diosi and Z. Krulich using Mayang's Free Textures library: http://mayang.com/textures/. We thank to J. Gemrot, R. Kadlec, J. Tomek, R. Pibil, and three anonymous referees for their comments on the paper.

References

1. Kelleher, C.: Motivating Programming: Using storytelling to make computer programming attractive to middle school girls. PhD thesis. Carnegie Mellon University, School of Computer Science, Technical Report CMU-CS-06-171 (2006)
2. Wilensky, U.: NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University (1999), http://ccl.northwestern.edu/netlogo/ (6.6.2011)
3. Gemrot, J., Kadlec, R., Bída, M., Burkert, O., Píbil, R., Havlíček, J., Zemčák, L., Šimlovič, J., Vansa, R., Štolba, M., Plch, T., Brom, C.: Pogamut 3 Can Assist Developers in Building AI (Not Only) for Their Videogame Agents. In: Dignum, F., Bradshaw, J., Silverman, B., van Doesburg, W. (eds.) Agents for Games and Simulations. LNCS, vol. 5920, pp. 1–15. Springer, Heidelberg (2009), http://pogamut.cuni.cz (6.6.2011)
4. Brom, C., Gemrot, J., Burkert, O., Kadlec, R., Bída, M.: 3D Immersion in Virtual Agents Education. In: Spierling, U., Szilas, N. (eds.) ICIDS 2008. LNCS, vol. 5334, pp. 59–70. Springer, Heidelberg (2008)
5. Artifical Minds for Intelligent Systems (AMIS): StoryFactory – a tool for creating machinimas in UnrealEngine2RuntimeDemo (in Czech), http://storyfactory.cz/ (6.6.2011)
6. Popelova, M.: Knihovna steering technik pro virtualni agenty. Bachelor thesis. Charles University in Prague (in czech) (2011), http://amis.mff.cuni.cz/emohawk/ (6.6.2011)
7. Gebhard, P.: ALMA - A Layered Model of Affect. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), Utrecht, pp. 29–36 (2005), http://www.dfki.de/~gebhard/alma/ (6.6.2011)
8. Ortony, A., Clore, G.L., Collins, A.: The cognitive structure of emotions. Cambridge University Press, Cambridge (1988)
9. Gemrot, J., Brom, C., Bryson, J., Bida, M.: How to compare usability of techniques for the specification of virtual agents' behavior? An experimental pilot study with human subjects. In: Proc. Agents for Education, Games, and Simulation, AAMAS Workshop (2011)
10. Rabin, S. (ed.): AI Game Programming Wisdom I-IV, Charles River Media (2002-8)
11. Bryson, J.J.: Inteligence by design: Principles of Modularity and Coordination for Engineering Complex Adaptive Agent. PhD thesis, MIT, Department of EECS, Cambridge, MA (2001)
12. Aylett, R.S., Louchart, S., Dias, J., Paiva, A.C.R., Vala, M.: FearNot! - An Experiment in Emergent Narrative. In: Panayiotopoulos, T., Gratch, J., Aylett, R.S., Ballin, D., Olivier, P., Rist, T. (eds.) IVA 2005. LNCS (LNAI), vol. 3661, pp. 305–316. Springer, Heidelberg (2005)
13. Lim, M., Dias, Y., Aylett, J., Paiva, R., Creating, A.: Creating adaptive affective autonomous NPCs. In: Autonomous Agents and Multi-Agent Systems, pp. 1–25. Springer, Heidelberg (2010)
14. Porteous, J., Cavazza, M., Charles, F.: Applying planning to interactive storytelling: Narrative control using state constraints. ACM Trans. Intell. Syst. Technol. 1(2) (2010)
15. Mateas M.: Interactive Drama, Art and Artificial Intelligence. PhD thesis. Department of Computer Science, Carnegie Mellon University (2002)