Designing and Testing Believable Agents for Independent Computer Games

JONATHAN HISE KALDMA

KTH Computer Science and Communication

Master of Science Thesis
Stockholm, Sweden 2009
Designing and Testing Believable Agents for Independent Computer Games

J O N A T H A N  H I S E  K A L D M A

Master’s Thesis in Human Computer Interaction (30 ECTS credits) at the School of Engineering and Business Management Royal Institute of Technology year 2009
Supervisor at CSC was Sinna Lindquist
Examiner was Lars Kjell Dahl

TRITA-CSC-E 2009:054
ISRN-KTH/CSC/E--09/054--SE
ISSN-1653-5715

Royal Institute of Technology
School of Computer Science and Communication

KTH CSC
SE-100 44 Stockholm, Sweden
URL: www.csc.kth.se
Abstract

The defining feature of the computer as a medium is interactivity. Yet, today’s computer games are remarkably non-interactive when it comes to the story and the characters in the game. One of the reasons is that it is difficult to create computer game characters that are both interactive and believable as characters. One way of doing it is create believable agents—autonomous agents whose main task is to be believable as characters. Believable agents should be of potential interest to independent game developers who are looking for new ways of furthering the computer game as an art form. This thesis explores how an independent game developer can design and test a framework for believable agents. During the project a prototype has been developed according to the design methodology of starting with a system whose behavior is completely random, and then iteratively introducing mechanisms for controlling randomness. The prototype has then been tested with users to evaluate its believability in the same way that computer systems are normally tested to evaluate their usability. Both the design methodology of working iteratively from randomness and the idea of testing believability in the same way as usability were found to work well.
Sammanfattning

**Att designa och testa trovärdiga agenter för indiespel**

## Contents

1. Introduction .......................................................................................................................... 1
2. Theory and method .............................................................................................................. 3
   - What is believability? ........................................................................................................ 3
   - Designing for believability ............................................................................................ 4
   - Testing believability ....................................................................................................... 6
3. Learning from existing believable agents ........................................................................... 8
   - Storytron .......................................................................................................................... 8
   - Façade .............................................................................................................................. 9
   - Drama Princess ................................................................................................................ 10
4. Designing a believable agent for independent games ...................................................... 11
   - Requirements ................................................................................................................... 11
   - Characters, actions and moods ....................................................................................... 11
   - Iterating from randomness ............................................................................................. 13
5. Testing the believable agent ............................................................................................. 15
   - The prototype .................................................................................................................. 15
   - Observing and interviewing ............................................................................................ 16
   - Feedback ........................................................................................................................ 17
6. Conclusion ......................................................................................................................... 19
7. Discussion ........................................................................................................................... 20
References ................................................................................................................................ 21
1. Introduction

The video games industry is one of the world’s largest entertainment industries. A video game production can easily rival a film production, with a budget of millions of dollars and a development period of several years. At the same time, the games produced by the industry are often lacking in artistic merit compared to what is produced in other entertainment industries. The most remarkable shortcoming of video games is how bad they are at integrating what the computer is good at—interactivity—with what other art forms such as literature, theatre and cinema are good at—story.

This has given rise to the field of interactive storytelling, which tries to use artificial intelligence to solve the problem of combining story with interactivity. Research in interactive storytelling has identified two hurdles that need to be overcome to create stories that are interactive: the first is the problem of creating characters that can think and act for themselves; the second is the problem of creating systems that can shape the interaction over time into a dramatically interesting whole. Most of the focus has been put on the first problem: creating characters that can think and act for themselves. This has led to the development of so-called believable agents.

A believable agent is an autonomous artificial intelligence whose main task isn’t to be intelligent, but to be believable—to convince the user that it has personality, feelings and goals. Unlike traditional autonomous agents, a believable agent doesn’t have to solve problems or accomplish goals effectively. All it has to do is interact with the player and other characters without breaking the player’s suspension of disbelief. And that’s not the easiest thing in the world.

Many systems for believable agents have been created by researchers in interactive storytelling. However, these systems have yet to make it into any major computer games. When AI is used in the games industry, it is instead used to make strategic or tactical decisions for the computer—to provide the player with an obstacle or competition, not to make the player believe in the characters. Perhaps that is understandable; with millions of dollars at stake, publishers aren’t too keen on experimenting with unproven technology like believable agents and high concepts like interactive storytelling.

But while the games industry continues to churn out safe bets, in the last couple of years a vibrant movement of indie games has begun to show up. These independent developers are people working in their spare time or on small budgets, who create games not to make money, but because they want to. They often have fresh and innovative perspectives not only on what games are, but on what games could possibly become. Titles such as Braid, Flower, and The Path all show that games have more to offer than just shooting things and collecting stuff.

If anyone can put the believable agent technology developed in the field of interactive storytelling to good use, it’s these independent game developers. They have the curiosity, they have the guts, and they have the desire for experimentation. The only problem is that believable agents are often prohibitively complex. Many of the systems were developed in
research labs as proofs of concept, and they were never released for public use. As such, an independent developer interested in believable agents would have to develop their own system from scratch. And trying to compete with the big boys of game development is hard enough already, even without trying to design and implement your own AI.

This thesis will look into how indie game developers can do it anyway. It will do so by trying to see what the indie game developer can learn from the field of Human-Computer Interaction (HCI). HCI has a long history of studying how we can successfully design for the interaction between humans and computers, and how we can evaluate the interaction to see that it works as we intended. Yet, even though computer games most definitely are a form of human-computer interaction, the concepts and methods from HCI aren't very common in game development, neither in the industry, nor among the independents. By applying concepts and methods from HCI, I thus hope to answer two questions: How can an independent game developer design and develop a believable agent? And how can that same developer evaluate the believable agent to see that it is actually believable?
2. Theory and method

Which is more important, plot or character? Let us trade the sensitive brooding Hamlet for a pleasure-loving prince, whose one reason for living is the privileges his princehood affords him. Would he avenge his father’s death? Hardly. He would turn the tragedy into comedy.

LAJOS EGRI, The Art of Dramatic Writing, p. 94

When trying to answer something as broad as how an independent game developer can design and evaluate believable agents there is really only one way: to try. A master’s thesis is written under circumstances very similar to those that the independent game developer faces: a team of one person, a time frame of 5–6 months, and no budget to speak of. Thus, this thesis has essentially been a case study in designing and testing a believable agent. But before we go into how it was done, we should look closer at the concept of believability, since it is central to believable agents. What is it? How can you design for it? And how can you measure it?

What is believability?

In the traditional narrative arts such as literature, theatre and cinema, one of the most important parts of a story is its characters. For a film, play or novel to be good, it isn’t enough that it has a captivating plot, an interesting setting, and a resounding theme—it must also have believable characters.

The question of what makes a character believable is one of the topics that have been heavily discussed throughout the history of the arts. From Aristotle’s Poetics to McKee’s screenwriting classic Story, novelists, playwrights and screenwriters have discussed and debated what makes a character not just a character, but a great character. That is not to say that there is a single definition of what makes a good character in the arts. As an example, in his classic text on playwriting The Art of Dramatic Writing, Egri says that characters have three dimensions that the dramatist must know: physiology, sociology, psychology (Egri 1960, 33). On the other hand, screenwriting guru McKee argues that the phrase “a three-dimensional character” means that a character is filled with contradictions (McKee 1999, 378).

Given this difference of opinion among traditional storytelling scholars, it should come as no surprise that there isn’t a clear definition of believability among scholars of interactive storytelling either. Loyall, who was among the first to use the term “believable agent”, derived seven criteria for believability from the traditional arts and from artificial intelligence research: personality, emotion, self motivation, change, social relationships, consistency and illusion of life (Loyall 1997). Another view is presented by Magerko, who means that believability is a sum of how well the character matches the player’s expectations on the character (Magerko 2007).
Despite this disagreement on what believability actually is, both in traditional and interactive characters, most of these definitions have one thing in common: they view believability as an intrinsic trait of the character. According to Egri, McKee and Loyall, believability is something that the author puts into a character when it is created, and once it’s there, the character is believable. This may make perfect sense in the world of literature, drama and screenwriting, where the communication of the medium is one-way. Egri and McKee are after all giving advice to authors whose characters won’t have to interact with the audience. But in a two-way medium such as computer games, this way of looking at believability misses an important part of the equation—the user.

Instead, I would like to propose that believability be viewed as a quality not of the believable agent, but of the interaction between the agent and the user. This is more in line with Magerko’s definition, which takes the player’s expectations into account. According to Magerko (2007, 80) “whatever ‘believability’ may be, a performance is lacking in it if that performance promises something it cannot deliver.” Thus, believability depends not only on the agent, but on the context as well. A believable agent that is believable in one game may be completely unbelievable in another game.

**Designing for believability**

The main thing that separates the independent game developer from the games industry developer is resources. The independent game developer has a team of one or two people, a development period of 1–2 years and no budget to speak of, while the industry game development studio has teams of 30–200 people, a development period of 2–4 years and a budget of several million dollars. This means that the independent game developer must not only design within constraints, but also for constraints. The games industry can afford to solve problems with systems that are both hard to develop and hard to use; the independent developer must instead design systems that are both easy to develop and easy to use. This naturally means cutting corners, but it also means being smart about design.

**Techniques in HCI**

In HCI, there has been much research into methods and practices for designing and developing computer systems so that they are usable. These methods and practices range from flexible techniques that designers can use as they see fit, to structured frameworks that not only tell you what to do, but when to do it and how to do it. For the indie game developer it is the techniques that are of most interest, since any framework will likely have too much overhead to be worth it. Thus, we will look into some of the most common techniques here: user-involvement, prototyping, iteration and expert evaluation.

*User-involvement*, or user-centeredness, is the idea that users and their needs should be the driving force behind the design from the beginning. To do user-centered design, you have to study the users and the contexts in which they will use the system; you have to capture and design for the users’ characteristics and limitations; you have to consult the users throughout the development and take their input into account; and you have to take all design decisions within the context of the users, their work and their environment (Preece, Rogers and Sharp 2002).
Prototyping is the practice of making a simple representation of the system to test certain aspects of it before it is developed in full. Prototypes can be either low-fidelity, e.g. simple drawings on a paper, or high-fidelity, e.g. a test version of the software on a computer. When a high-fidelity prototype is further developed into the final product it’s called evolutionary prototyping. When it is thrown away, and the final product is developed from scratch, it’s called throwaway prototyping. (Preece, Rogers and Sharp 2002)

 Iteration is the design and development method of working in cycles; making a first version, changing it based on feedback, making a second version, changing that based on feedback, etc. While this is common practice in the arts, it hasn’t always been so in systems development, where computer systems have often been specified in excruciating detail before they are actually built.

Expert evaluation means simply asking experts what they think. However, since you don’t always have an expert at hand, you may instead have to use heuristics, or rules-of-thumb. Heuristics are lists of common best-practices that the system should adhere to. Perhaps the best known list of heuristics is Jacob Nielsen’s Ten Usability Heuristics (Nielsen 2005).

Applying HCI techniques to believable agents
Which of these techniques then are appropriate to an indie game developer designing a believable agent? The resource constraints are certainly important to consider—e.g. an indie developer can’t afford extensive studies of people playing computer games in their homes. But the main thing to keep in mind is the difference between computer programs and computer games. The practices in HCI have all grown from the design of systems that are meant to “support people in their everyday and working lives” (Preece, Rogers and Sharp 2002, 6). The aims of these systems are to be useful and usable. But computer games aren’t meant to be useful, they are meant to entertain. And believable agents aren’t meant to be usable, they are meant to be believable. As such, not all the techniques that are good for designing usable computer systems need be good for designing believable agents.

User-involvement is one such technique. While an essential tenet of HCI, user-involvement doesn’t fit the process of designing a computer game or a believable agent too well. To let the design process be driven by the needs of the user, the user has to have specific needs that the system can fulfill. But computer games don’t fulfill specific needs. They fulfill general needs, such as the need for relaxation, or the need for excitement. Developing an indie computer game is more akin to writing a book or screenplay; the process starts and ends with the writer’s vision, not with the needs of the audience. Thus, for this project I chose not to have a user-centered approach to design.

While user-involvement may not be applicable to believable agents, prototyping certainly is. From the beginning of the project, it was clear that I needed to make a prototype not only of the believable agent, but of an actual computer game where the user could interact with the believable agent. The questions were whether that prototype should be low-fidelity or high-fidelity, and whether it should be evolutionary or throw-away. It seemed hard to make a low-fidelity prototype which could accurately test what I wanted to test—believability—so I opted instead for making a high-fidelity prototype from the start. Also, since the very idea was to make something that could be used by an indie game developer,
throwing away the prototype at the end wasn’t an option. Instead, I made it a point to
develop the prototype to such standards that it could be further developed into a complete
game. It should be noted that this approach has a certain risk associated with it—if the
prototype hadn’t worked, I would have spent valuable time not only on building the
prototype, but on building it well.

Iteration is another technique which seemed appropriate to an exploratory project such as
this. When working iteratively it’s easy to change course if you discover a better way to do
something than what you had initially planned. Furthermore, when working in a one-man
team, you have none of the organizational problems that a larger organization may have
with iteration. However, just because you work iteratively doesn’t mean you shouldn’t
have structure to your work process. Thus, I chose to use a light-weight process inspired
by the agile software development method Scrum. I worked in two-week iterations, and at
the start of each iteration I made a list of features that should be completed before the end
of the iteration. During the iteration, this list was essentially fixed, and only minimal
changes to it were allowed. To allow the process to be completely iterative, I also chose to
develop the believable agent and the computer game in parallel. The point was to allow
the design of the computer game to inform the design of the believable agent, and vice
versa.

Finally, expert evaluation is a technique which I chose not to use. The idea is certainly
interesting—to let experts evaluate the believable agents, or to use heuristics for
evaluating believability—but since no clear consensus exists on what believability is, it’s
unclear who the experts would be, and what the heuristics would consist of.

Testing believability

As mentioned above, believability is something that the user experiences when interacting
with a believable agent. In this way believability is much like the concept of usability in
HCI. Usability isn’t a quality of the system itself, such as green or fast, but a quality of the
interaction between the user and the system. A computer system is deemed usable if it is
effective to use, efficient to use, safe to use, has good utility, is easy to learn, and is easy to
remember how to use (Preece, Rogers and Sharp 2002). How then do you evaluate if a
system is usable? The most common way of doing this is by a technique not mentioned so
far: usability testing.

Usability testing is the practice of testing a prototype or developed system with actual
users to see that it works as intended. It relies on a combination of techniques, including
observing users as they use the system, measuring their performance at certain tasks
under controlled conditions, and doing questionnaires and interviews with them about
their experience (Preece, Rogers and Sharp 2002). Usability testing can either be
quantitative or qualitative. Quantitative testing uses objective quantities, such as number
of errors per minute, or minutes to complete a task, to measure some aspect of usability.
Qualitative testing instead aims not to measure, but to understand and gain insight into
what works and what doesn’t, and why some things work and some things don’t.
If usability can be evaluated this way, it should be possible with believability as well. But which approach should be taken, the quantitative or the qualitative one? Magerko suggests that believability can be measured quantitatively as a “function of the sum across modalities of a character’s adherence to the set of relevant constraints” (Magerko 2007, 81) (emphasis in original). However, he provides no explanation of how his measure should be practically applied, and even concedes that it probably raises more questions than it answers (Magerko 2007, 82). Indeed, it seems strange to think that believability, which is essentially an artistic quality, could be measured. No-one in their right mind would attempt to measure how beautiful a painting is, or how dramatic a play is.

It seems then, that the approach most suited to testing believability is the qualitative one. There are two ways in which this is done in usability testing: observations and interviews.

Observing users lets the designer see what happens when the system actually hits the road. Does the map and the terrain match? When observing how users interact with the system, the designer can discover that things that are obvious to the designer are far from obvious to the user. There are different approaches to observations. They can be anything from “quick and dirty” observations done without much formality, to field studies and observations in controlled environments (Preece, Rogers and Sharp 2002). I have chosen to use an approach leaning towards the “quick and dirty.” With no budget and little time, it isn’t really possible to do a study in a controlled environment. While this may certainly make the results less reliable, it should also make it easier to get as much feedback as possible. After all, the point of evaluating believability is really to aid the game developer in the creative process, not to establish general truths.

Interviews often go hand in hand with observations. When you’re observing users, you might as well talk to them. Interviews let the developer gain an understanding not only of the problems with the system, but of the user and their thoughts. Interviews can be structured, semi-structured or open, depending on the degree to which the interviewer follows a predetermined set of questions (Preece, Rogers and Sharp 2002). I chose to use a semi-structured approach, i.e. having a list of questions to which I was seeking answers, but not necessarily asking the questions in order or word for word. This approach naturally has lower reliability than a completely structured approach, but in return it lets the interviews take unexpected turns, giving the interviewer richer data.
3. Learning from existing believable agents

Look you, before he cries ‘Zaïre vous pleurez,’ or ‘Vous y serez ma fille,’ the actor has listened over and over again to his own voice. At the very moment when he touches your heart he is listening to his own voice; his talent depends not, as you think, upon feeling, but upon rendering so exactly the outward signs of feeling, that you fall into the trap.

DENIS DIDEROT, *The Paradox of Acting*, p. 19

The point of this thesis is to develop an architecture for believable agents within time and resource constraints similar to those of an independent game production. And the only way to do that is to stand on the shoulders of giants.

Since research into interactive storytelling began in the 1990s there have been several attempts at creating systems for interactive storytelling and believable agents. Good sources of knowledge on these systems are the papers produced by The Oz Group at Carnegie Mellon (OZ Project Publications n.d.), and the proceedings from the AAAI symposia on Intelligent Narratives and Intelligent Narrative Technologies (Mateas and Sengers 1999) (Magerko and Riedl 2007) (Louchart, Mehta and Roberts 2009). In addition, there are systems that are not part of the research tradition of believable agents, but are still of great interest, such as Storytron (Crawford 2005) and Drama Princess (Samyn 2006-2007).

Naturally, for this project I couldn’t study all of these systems. Instead, I selected three systems that I found particularly interesting. They are all clearly different from each other, and so should be able to teach different things. Also, they have all been used to create an actual game or interactive drama, which isn’t all too common in the field of interactive storytelling, where much work is otherwise on the prototype level. The systems I have chosen to look closer at are Storytron, *Façade* and Drama Princess.

Storytron

Storytron began when computer game guru Chris Crawford decided to leave the computer game industry in 1992 to pursue what he thought would be the future of computer entertainment: interactive storytelling. Then dubbed Erasmatron, Storytron has now been in development for over a decade, and is finally starting to become finished. The first finished game, or “storyworld”, called *Balance of Power 21st Century*, has just been released to the public.

Storytron is an attempt at creating a system not just for believable agents, but for interactive storytelling in general. It is based on Crawford’s definition of interactivity as “a cyclic process between two or more active agents in which each agent alternately listens, thinks, and speaks,” (Crawford 2005, 29) where listening, thinking and speaking are
meant metaphorically. From this definition Crawford deduces the concept of the verb as the fundamental unit of interactive storytelling. A verb is something that the characters in an interactive story can do, and the main task of the interactive storytelling system then becomes to determine how the characters in the world react when a character does a verb.

Storytron does this using roles and inclination formulae. Roles are the relations characters have to the verb, e.g. “subject,” “direct object,” “girlfriend of subject,” and “father of direct object”. The first thing that the system does when determining the reaction for a character is to determine which role fits which character. The most common role will be the direct object role, i.e. the character that the action is done to. Other common roles will be characters who have close relations to the subject or direct object, and who would want to know and react to things that they do or that happen to them.

For each role, a verb has a number of possible reactions (i.e. other verbs) listed, and each of those reactions has an inclination formula. Out of these possible reactions, the system chooses one by evaluating the inclination formulae for all of them, and picking the one with the highest inclination. The inclination formulae are mathematical expressions that use the psychological models of the characters to calculate some kind of value, e.g. Inclination[Revenge] = Hate[Subject, Object] – Virtue[Subject] means that the more the subject hates the direct object the higher inclination it has for revenge, but the higher the virtue of the subject the lower inclination it has for revenge.

The main strength of Crawford’s design is that the game developer, or “storybuilder” as they are called, has complete control over every nuance of what will happen in the game. The main drawback is that the system isn’t built to integrate with common game engines; the user interacts with the system by choosing actions from drop down menus. Also the amount of authoring that is required just to build a simple storyworld, let alone one with many possibilities, seems prohibitive. For each verb, a number of roles must be defined, and for each role a number of reactions must be defined, each with its own finely tuned inclination formula.

**Façade**

*Façade* isn’t really a system for believable agents, but rather a game that uses them. It was created by researchers Andrew Stern and Michael Mateas, and was the only actual interactive drama to come out of the research project that coined the term "believable agent": The Oz Project at Carnegie Mellon University.

The believable agent technology behind *Façade* is called ABL, which is short for “A Behavior Language,” and it’s based on an earlier system developed by the Oz Project called Hap. In ABL, the actions and personality of each character is defined with goals and behaviors. A goal is something that the character wants to do, e.g. “stop fight,” and a behavior is a list of physical actions that the character has to take to accomplish that goal, e.g. “get between combatants” and “tell combatants to stop fighting.” A goal can have multiple behaviors that can satisfy it, and a behavior can include subgoals that need to be satisfied for the behavior to be fulfilled. Together these goals and behaviors create a tree structure called the Active Behavior Tree. Each character has its own ABT that can change
over time, and it is by evaluating the behavior tree that the system determines what the character should do next.

While the structure of ABL makes it possible to create characters that pursue goals realistically, it is unclear from the architecture how the game author should use the system to create interaction. It seems that the system is more of a system for autonomous agents than for interactive autonomous characters, and that any interactivity that may arise from the system is only indirectly created by the game author. Furthermore, behaviors are written in their own programming language, adding to the burden both of implementing the system in a game engine and of authoring behaviors with the system.

**Drama Princess**

Drama Princess is a project by independent game studio Tale of Tales, made up of artists Michaël Samyn and Auriea Harvey. The aim of the project was to build an autonomous character for interactive storytelling that they could use in their game productions. The development of the system was made open to the public on a special blog by Samyn, and the system made its debut in the critically acclaimed horror game *The Path*.

What separates Drama Princess from Storytron and *Façade* is the approach; Samyn and Harvey have an outside-in approach, trying to make the user believe that the agent has a mind, rather than actually making the agent have a mind. Samyn makes the comparison with acting, where there is the Diderot school, according to which the actor should only imitate the emotions of the part he is playing, and method acting, according to which the actor should try to actually feel the emotions of the part he is playing. (Samyn 2006-2007) Drama Princess is thus akin to Diderot acting, while AI-based systems such as Storytron and *Façade* are akin to method acting.

Because Drama Princess isn’t out to simulate what the character actually thinks, it relies heavily on the idea of using randomness rather than simulation to choose what the character should do. All objects in the world, including characters, have lists of actions that can be done to them. A character then takes all these opportunities and puts them in a list. It then uses a couple of filters to culls that list, e.g. if the object is close enough and if the object has good relations to the character. From this culled list the character then randomly chooses an action to perform.

Drama Princess shows that it is indeed possible to create a believable agent system as an independent game developer. The system also has a very innovative way of minimizing authoring: rather than spend a lot of time carefully crafting rules for how behaviors should be selected, Samyn and Harvey have the approach that “good enough is good enough” and let the system randomize between behaviors instead. The downside of this however is that the game developer has little direct control of what happens.
4. Designing a believable agent for independent games

It seems that perfection is attained not when there is nothing more to add, but when there is nothing more to take away.

ANTOINE DE SAINT-EXUPÉRY, *Wind, Sand and Stars*, p. 31

Standing on the shoulders of giants is a good way to start, but it won’t get you all the way. This chapter will describe how I designed and developed a believable agent framework for independent games.

Requirements

The purpose of the framework is to allow an independent game developer to create believable autonomous characters. As such, the system not only had to to enable interaction between characters that was believable, it also had to be usable and flexible enough to create a real game.

The framework had to:

- **Integrate with an affordable game engine**
  As an indie developer, you don’t have the resources to develop your own game engine. Luckily, there are several affordable game engines available on the market, e.g. Torque 3D and Unity. For this study, I chose to use the Unity game engine, as it allows scripting in C#, an industry standard language, and has an Integrated Development Environment (IDE) that focuses on usability and flexibility.

- **Be easy to use for the developer**
  I decided at the start of the project that I wouldn’t have the time to learn a new programming language, or write a parser for such a language. All scripting in the framework had to be done in C#. Also, since an evolutionary approach was chosen, the framework should support actually making a complete game with it. This meant that it had to integrate well with the Unity IDE.

- **Be flexible**
  Another important requirement for being able to use the framework to create a complete game was that it was flexible. I wanted the framework to be easy to use in the future, and not require “tinkering under the hood” to accomplish fundamentally different types of behavior.

Characters, actions and moods

From the study of existing believable agents, it seemed clear that any system for believable agents also had to be a system for creating interaction. Storytron, ABL and Drama Princess are all systems for defining what a character can do, just as much as they are systems for
defining what a character is. It also seemed clear that the system had to let the characters express emotions in order to be believable. The basic units of the framework therefore were chosen to be characters, actions and moods.

Characters
Anything in the game that can perform actions and react to actions is a character. Most likely, this means that all the characters in the story will also be characters in the framework. However, it is also possible to make e.g. an item into character, if you want that item to be able to act and react.

Unity uses a component architecture where an object in the game is defined by adding components to it. For example, to add animation to an object, you add the animation component to that object. To make the framework as flexible as possible, a character in the framework is just another such component, meaning that any object in the game engine can be made into a character.

Actions
I took a cue from Crawford by making the action—or as he calls it, the verb—the basic unit of interaction. Anything that can be done by the characters in the game is defined as an action. This leads to symmetry in the interaction: the user can do the same things to the other characters in the game that the other characters can do to the user. The concept is also very simple for a game developer to grasp and use—anything that one character can do to another character in the game is an action. Examples of actions are kiss, punch in face, hug, etc.

To make the framework as flexible as possible each action has a script that is run when a character performs the action. This script is written in C#, just like all other scripts in the game engine, and has access to the whole game engine. This allows the game developer to do anything with it—play an animation, play a sound, shoot a projectile at another character, give an item to another character, run away from another character, etc.

Furthermore, to enable re-use of code, actions can share scripts, yet have different inputs to those scripts. This means that generalized scripts, such as playing an animation and a sound effect, can be reused between actions (even if the actions play different animation and sound effects).

Moods
Just being able to perform actions is not enough to make a character believable. The character must also be able to express emotions. Here, I took the opposite solution to that of Storytron and Façade, and made no distinction between how a character feels and how the character expresses that feeling onwards. This was very much inspired by the Drama Princess idea of being like the Diderot actor: showing feeling, but not actually feeling anything. Thus, in my framework, there is only one representation of a characters’ feelings, and that is its mood.

A mood is a way that a character looks and sounds like when it is performing actions, e.g. angry, happy, sad, blushing, etc. To make the framework as simple as possible to use, each
character at any moment has exactly one mood. There is no internal representation of a character's emotions other than its mood. Moods, just like actions, have their own scripts that determine how they play out.

Iterating from randomness

Thus far the design decisions haven't been that hard. After all, any system for interactive characters needs some representation of characters, behaviors and emotions. When designing how characters should react however, decisions had to be made.

The main difference between Storytron and Façade on the one hand, and Drama Princess on the other, is the way that behaviors are selected. Storytron and Façade both base their selection on some type of simulation. Drama Princess bases its selection on randomness. For this project, I chose to go the Drama Princess way, and use randomness. The idea was to start with a system that is completely random, and then iteratively introduce mechanisms for controlling randomness.

Complete randomness

The first iteration of the framework was completely random. When one character performed an action on another character, that character chose a reaction at random from all the actions in the game. This naturally led to reactions that were absolutely non sequitur, such as one character waving at another character, and that character walking off.

Randomness by contexts

The second iteration organized actions into contexts. The idea was that a context is a group of actions that could all believably follow from each other. For example, if there is a break-up context, the actions in it might be say it's over, cry, blame the other, blame yourself, and say you'll change. Any of those actions could quite believably follow from any other. While there are certainly combinations that might test the limits of believability, none of them would be completely unbelievable.

The second iteration also introduced moods for the characters. These were organized into the same contexts as the actions, such that the break-up context might include the moods sad, angry, despairing, and indifferent. When a character performed an action on another character, that character chose both a mood and a reaction at random from the same context as the first action.

To enable the current context to change, I made it possible for actions to belong to several contexts. The idea was that most actions would only belong to one context, but some actions would be “portal actions” that could cause a switch from one context to another. For example, the action say you'll change could cause the context to change from break-up to reconciliation, but it could also have no effect on the context, which would then still be break-up. Thus, when an action was part of two or more contexts, the character randomly chose one of the contexts as the context within which to find a mood and reaction. The concept is illustrated in Figure 1.
This design worked much better than the complete randomness of the first iteration, but it still didn't feel very believable. Non-sequiturs could still appear, such as when the chosen action and the chosen mood didn't match. A character might for example curse and be happy at the same time.

Randomness by context and conditions
The third iteration introduced conditions to actions. These were Boolean expressions that were evaluated for the specific characters involved in the reaction. For an action to be available for randomization it not only had to be in the right context, but its conditions also had to be true. For example, this made it possible to make the action *curse* available only if the character's mood is *angry*, or not make it available if the other character's mood is *sad*.

This design was the first that I felt was successful. The interaction between the characters was interesting and any complete non-sequiturs had been ruled out. As such, I decided to be content with the design, and evaluate it.
5. Testing the believable agent

The experience of interactivity is a thresholdy phenomenon, and it is also highly context-dependent. The search for a definition of interactivity diverts our attention from the real issue: How can people participate as agents within representational contexts? Actors know a lot about that, and so do children playing make-believe. Buried within us in our deepest playful instincts, are the most profound and intimate sources of knowledge about interactive representations.

BRENDA LAUREL, *Computers as Theatre*, p. 21

To evaluate the believable agent the idea was to test it with actual users, much in the same manner as usability testing is done in HCI. The idea was both to see whether users actually found the characters to be believable, but also to find weaknesses in the framework that could be improved.

The prototype

The game prototype that I created in parallel with the believable agent framework was by necessity quite small. It consisted of a small island upon which there were three small robots as shown in Figure 2. One of the robots was controlled by the player. Clicking on the ground made the player robot move to the spot that the user clicked on, and clicking on another robot brought up a menu of the actions that the player robot could do to the other robot. The player could also click on items on the ground to pick them up, and double click on the ground to put them down.

*Figure 2* Screenshot of prototype
There were seven actions and three moods available to the robots. They were divided into two contexts, friendly and hostile. The actions are shown below in Table 1, and the moods in Table 2.

<table>
<thead>
<tr>
<th>Friendly</th>
<th>Hostile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blow bubbles</td>
<td>Curse</td>
</tr>
<tr>
<td>Jump</td>
<td>Run away</td>
</tr>
<tr>
<td>Poke</td>
<td>Shake head</td>
</tr>
<tr>
<td>Wave</td>
<td></td>
</tr>
<tr>
<td>Run away</td>
<td></td>
</tr>
</tbody>
</table>

Table 1 Actions according to context

<table>
<thead>
<tr>
<th>Friendly</th>
<th>Hostile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Happy</td>
<td>Angry</td>
</tr>
<tr>
<td></td>
<td>Sad</td>
</tr>
</tbody>
</table>

Table 2 Moods according to context

The environment around the robots was a small island with a dock, some weeds, a couple of stones on the ground, and a building that couldn’t be entered. Since the purpose of the prototype was to test the interaction between the robots, there wasn’t anything to do on the island except interact with the other robots. There was only one item that the player can pick up: a “bubble blower” which let the robot blow soap bubbles at another robot.

**Observing and interviewing**

As discussed in the chapter on theory and method, my approach to testing believability was a qualitative one based on “quick and dirty” observation of usage and semi-structured interviews.

The test procedure was as follows: The tester was first presented with a consent form that explained the purpose of the tests, how the tests would be conducted and their rights as participants in the test. They were then shown the game prototype and told to “explore it.” Instructions that explained how to control the main character and how to interact with other characters were automatically shown at the start of the game. While the tester played the prototype, I observed them quietly, only answering questions if they got stuck or if they had completely misunderstood how to interact with the game. I didn’t set any time limits or give them any tasks. When they felt that they had explored the game prototype enough (usually after 5–10 minutes), I interviewed them about their experience for another 5–10 minutes.

The interviews were based around a list of questions, which were adapted to the situation and the interviewee. I tried to determine whether the testers found the characters believable both by asking them so directly, and by asking them indirectly: Had the characters done anything that disrupted their suspension of disbelief? Had they seen any
patterns to the actions of the robots? And what was missing that would make the robots even more interesting?

The tests were conducted at a location of the tester’s choice with a laptop. I recruited the testers from my extended network, i.e. from friends of friends, co-workers of friends, family of friends, etc. To account for possible biases among the testers, I tried to find testers that covered as wide a range as possible in gender, age and previous experience with computer games. Still, I couldn’t test the game with an infinite amount of people. According to Nielsen (2000) when testing usability, testing with five users is enough to identify the major usability problems in a design. Instead of testing with more users, Nielsen recommends testing with a couple of users, and then testing with more users after the problems uncovered by the first tests have been addressed. Thus, I limited myself to seven testers. They were:

- Man, 55, almost never plays computer games
- Woman, 50, never plays computer games
- Man, 40, never plays computer games
- Man, 28, sometimes plays computer games
- Woman, 24, never plays computer games
- Boy, 8, often plays computer games
- Boy, 7, sometimes plays computer games

Feedback

The observations and interviews yielded quite clear results. All testers said that they found the characters believable and fun to interact with. Also, all testers said that they thought that the reactions of the other characters were appropriate. None of the testers thought that any of the characters had done anything that was unreasonable or broke suspension of disbelief, and several testers saw patterns to the reactions of the other robots, such as “when you poke a robot it runs away,” that were not explicitly built into the prototype. Most importantly however, several testers expressed, either during the test or during the interview, that they attributed personality or volition to the robots, such as:

- “The red one was more easily scared than the blue one.”
- “After a while they got tired of you and went away.”
- “He got angry because you blew bubbles at his face!”

This confirms that the randomized design approach inspired by Drama Princess was a good one. People are indeed willing to see reasons behind the behavior of interactive characters even if those behaviors aren’t the results of an intricate simulation.

The characters should take more initiative

The framework isn’t without a fault though, as the testers also found things that hampered the believability of the characters. The worst such problem was that the non-player characters didn’t take any initiatives of their own. Two testers thought that this was boring and detracted from the believability of the characters. In the words of one tester:
“They didn’t move. They just stood there. It would have been more fun if they had moved around a bit as well.”

This was also confirmed by the observation of how most testers stopped interacting with the robots when they had tried each action once or twice. It’s likely to believe that had the non-player characters taken initiatives of their own and performed actions upon the player character independently, the testers would have responded to those actions.

**More actions are needed**

Another issue raised by several testers was that the interaction between the characters was too shallow and restricted. As one tester expressed it: “If you imagine that this is two humans, we have more, what to call it, we do more things. Like when we’re talking right now, we make gestures and things. Here it was a bit short. Blow bubbles and wave and get mad. If you compare it to a human we have a bigger registry of things to do.”

This problem was expected, as there were indeed very few actions available in the game prototype. Also, the actions weren’t very meaningful in a story sense. As such, this might not be as much a problem with the design of the believable agent framework, as a problem with the game prototype. When evolving the prototype into a real game, more actions will of course have to be added.

**Usability issues**

In addition to the feedback on believability, the tests also gave feedback on usability, even though they hadn’t been intended for that. Two testers clicked very quickly on the ground to move the robot, and were confused when this caused the robot to put down the bubble blower. Several testers tried repeatedly to make the robot move where it wasn’t possible; indicating that better feedback was needed on where the robot could move.
6. Conclusion

This thesis has been a case study in how an independent game developer can design and test a framework for believable agents. From this process two important lessons can be learned.

The first lesson is that it’s very much possible to design and develop believable agents with the scarce resources and time available to an independent game developer. I single-handedly developed the framework in this thesis in two months. The main design decision that allowed this was to *start with a system that is completely random, and then iteratively introduce mechanisms for controlling randomness*. Most existing believable agents have complex psychological or cognitive models for how the characters think. For the independent game developer this is completely unnecessary; the characters don’t really have to think at all—as long as the game developer has ways of filtering away completely unbelievable behaviors. Introducing these mechanisms iteratively lets the game developer minimize the amount of time spent developing features that won’t be used.

The second lesson is that believability can be tested in much the same way as usability is tested in the field of Human-Computer Interaction: by observing and interviewing users. The tests conducted in this thesis were short and mobile—10–20 minutes and a laptop. This “quick and dirty” approach to testing made it possible to test with people who otherwise didn’t play computer games, and get more feedback. The tests not only gave ideas how to improve the believable agents, but also on the usability and experience of the game in general.
7. Discussion

The results of this thesis should be of interest to any independent game developer who is curious of the technology of believable agents, or who wants to find new ways of creating more interactive stories in games. By using randomness to drive the characters in Drama Princess, Samyn and Harvey have come up with a new way of creating believable agents and demonstrated that it can work. This thesis takes that approach and applies it in an iterative manner. By starting from randomness and only introducing features to the believable agent as they are needed, the developer doesn’t have to spend time designing and implementing complex psychological or cognitive models unless they are truly necessary to make the characters believable. As shown, this can simplify the design process significantly.

The thesis also shows that it can be very fruitful to use techniques from HCI when designing believable agents for computer games. This intersection between interaction design, interactive storytelling and computer games could certainly be explored further. One idea that was mentioned but not delved into was to develop heuristics for believability, much like there are heuristics for usability. Another approach that wasn’t taken was to involve users in the design process from the very start. The prototype was only tested with users after three iterations. This could be criticized as being somewhat late. The “quick and dirty” nature of the tests certainly would have allowed me to do tests between iterations as well. Even though I as a designer felt that the prototype wasn’t believable then, testing it anyway might have provided insights that I didn’t come to on my own.
References


