Dynamic Interactive Storytelling for Computer Games Using AI Techniques

Simon Cooper, Dr Abdennour El Rhalibi, Prof. Madjid Merabti
School of Computing & Mathematical Sciences
Liverpool John Moores University
Byrom Street, Liverpool, L3 3AF
(+44) 0151 231 2282
s.cooper@2003.ljmu.ac.uk
a.elrhalibi@ljmu.ac.uk
m.merabti@ljmu.ac.uk

Dr Marc Price
BBC Research and Development
BBC Research, Kingswood Warren,
Tadworth, Surrey, KT20 6NP, UK
Marc.Price@bbc.co.uk

ABSTRACT
In this paper, we investigate the design and implementation of an interactive storytelling engine and the artificial intelligence techniques that will enable novel approaches to procedurally generating digital interactive storytelling for computer games.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Constraint Satisfaction, Control Methods, and Search – *Graph and tree search strategies, Heuristic methods, Plan execution, formation, and generation.*

General Terms
Algorithms, Design, Human Factors, Languages, Theory.

Keywords
Interactive Storytelling, Games, AI, Jess, CCP, Constraints Satisfaction Problem (CSP), Java Monkey Engine (jME), Homura.

1. INTRODUCTION
In this paper we investigate Artificial Intelligence techniques that will enable novel approaches to procedurally generating digital interactive storytelling for computer games. These techniques will provide new and immersive experiences for players.

Murray states, "The stories we tell reflect and determine how we think about ourselves and one another. A new medium of expression allows us to tell stories we could not tell before, to retell the age-old stories in new ways, to imagine ourselves as creatures of a parameterized world of multiple possibilities, to understand ourselves as authors of rules systems which drive behavior and shape our possibilities." [1]

The new medium referred to is that of computer games. Since 2004 when this statement was written, that medium has advanced technologically allowing rule based systems to be more complicated than ever, with the latest computers and games consoles having multiple threads and processors running in parallel. With this extra power and greater storage capacity games are becoming bigger and more epic projects with sprawling worlds, an array of characters and a more complex narrative. “In computer games characters aren’t presented, they are experienced.” [2]

Many systems based on rules and planning have been proposed in related literature, but these have many limitations and drawbacks. We propose to develop systems based on heuristics and meta-heuristics to provide a new view of emerging digital storytelling.

2. LITERATURE REVIEW
According to Spector [3] the features common to all good stories are change, pacing, compelling characters and subtext.

- Player’s actions “must change” themselves, the game world and the characters they meet.
- The pacing must vary to hold interest, with build-ups at the beginning and the end.
- Characters have to be interesting with their own personalities, goals, beliefs, needs and desires.
- The story should have a main plot subtext as well as the smaller sub-plots that make up the game.

Spector continues further by explaining five ways to approach storytelling in games: Rollercoaster, Retold, Sandbox, Shared Authorship and Procedural/Interactive Storytelling [3].

Rollercoaster storytelling uses a pre-determined narrative, which players traverse in a linear manner with no significant options or unique experiences.

Retold stories are abstract games with no story at all other than individual accounts of the game experience.

Sandbox storytelling allows the player to interact with toys and create their own story through the choices that they make.

Shared Authorship gives the player some freedom in a similar way to sandbox storytelling, but also has pre-determined goals as in rollercoaster. It is up to the player to decide the order they achieve the goals or to ignore them all together.

The last one, *procedural/Interactive Storytelling*, is the most interesting and is the focus of this research. It aims to give players more freedom to explore and develop relationships with the game characters and world through their choices.

One of the earliest games to use interactive storytelling was in fact a tabletop game Dungeons & Dragons, which was released in 1974 [4].

A player is appointed the role of games master (GM), whose job is to direct the players through the dungeon, whilst reacting to their actions, using the original story as a guide and filling in the blanks using their imagination and reasoning for a more entertaining experience. In computer games rule systems decide how to respond to player actions and replace the GM. These limited rules aren't capable of responding to any action in the same way a human can, so the freedom in the number of actions that can be performed at a given time is significantly less [4].

To create more interactive storytelling we need a larger entity controlling and adapting the story in accordance to the players' actions. Champagnat, Estraillier and Prigent suggest a narrative framework to simulate this activity. In this framework, they suggest having a set of agents in charge of evaluating the current state of the game, and one director agent, which "chooses a set of relevant actions to be executed" [5].

Cavazza, Charles and Mead use a hierarchical task network (HTN) planning method for interactive storytelling "as it is generally considered appropriate for knowledge-rich domains" [6]. The game characters' roles in the story can be represented as tasks, with a single HTN corresponding to several possible decompositions for the main task and the story being composed of situations that are the "cross-product" of the actors' roles [6]. The HTN is traversed executing primitive actions along the way and allowing backtracking if they fail. The fitting aspect of HTN planning is that it is based on forward searching "while being goal directed at the same time" [6], with the top task being a characters' main goal/objective (e.g. a cartoon villain would want to take over the world, but their sub goals would be to get a doomsday device and secret base in order to achieve this main objective).

Another technique generating interactive storytelling is described by Cai et al [7]. In this approach, a tool based on Goal Net is used to plan the story and Fuzzy Cognitive Maps are employed to analyze the user inputs and decide which path should be followed; they were selected due the fact that they act as "collection of the rules such that it not only concerns the relationships between the causes and effects, but also considers their relationships among the causes" [7]. Similar to the work described by Champagnat, Estraillier and Prigent, this approach implements an agent, modelled using the Goal Net tool, which will be in charge of presenting the story in accordance to the user actions [4][7]. In order to accomplish this, a controlling agent keeps information related to the states required to achieve a goal, and the relationships or transitions that connect those states [7].

Goal Net's ability to break goals down into simpler states make it useful for interactive storytelling [7][8]. Scenes can be divided into smaller segments creating multiple routes to the main plot subtext. It also has four different ways of connecting the small segments states (sequence, concurrency, choice and synchronization) allowing for flexible and more interesting transitions between the different states, which leads to the creation of more complex interlinking stories. Also important to note is that according to Cai et al, using Goal Net as planning tool instead of Hierarchical Task Network yields better results in terms of interactive storytelling, since it provides the ability to select scenes at real-time, according to the current context and user input [7].

The engine created by Cai et al consists of a knowledge database, where the scenes and their relations are stored, the fuzzy cognitive goal net engine, a container for the agents to be used, known as the Drama Manager, and a Multi Agent Development Environment (MADE) platform to implement the agents' system; using fuzzy cognitive maps, the engine will decide, at runtime, which scenes from the knowledge database should be loaded into the agents in order to better suit the path selected by the user [7][9].

Three main advantages can be distinguished from employing this method to create interactive storytelling [7][9]:

- All events are simplified into less complex scenes
- The engine can react not only to the user inputs but also to the actual state of the game, and look for the best path.
- Since all the events are loaded into the engine on real time, its performance is increased.

Mateas and Stern [10] created an experiment in interactive drama, where the player has to interact with two NPC's that are having marriage problems.

- The player's actions should have a significant influence on what events occur, which are left out, and how the drama ends.
- Drama manager monitors the simulation and adds and retracts procedures (behaviours) and discourse contexts by which Grace and Trip (the NPC's) operate.
- Plot is made up of dynamically sequenced *story beats.*

The Storytron platform [11] is one of the most complete examples of interactive storytelling. It is character based and uses a verb based action system. "It is centred on artistic works called storyworlds. Each storyworld is a universe of possible narratives. The technology is comprised of four parts: SWAT, the Storyteller, Deikto, Sappho, and the Storyengine". SWAT is used to create new storywords, the Storyteller is the software used to play storyworlds, Deikto is the English-like language used by players to create the sentences that drive their actions, Sappho is the scripting language the author uses to create a storyworld and the Storyengine drives the computer characters AI.

Despite the growing interest in Interactive Storytelling, there have been only a small number of implemented demonstrators and few have attempted at developing a re-usable Interactive Storytelling technology. In this research we will design such an Interactive Storytelling engine, which will be the result of a wide investigation on AI techniques, some of which are not used in previous research projects. The system will be based on the Java Monkey Engine (jME) game engine for its visualisation component, while the story generation component will implement different model based AI, using a combination of rules, constraints, verb based actions, triggers, states, communication architectures and personality models.

3. METHODOLOGY & IMPLEMENTATION

3.1 Engine Overview

Our engine will be called DISE – Digital Interactive Storytelling Engine. The engine will consist of several core components and technologies that were deemed necessary to make up a comprehensive interactive storytelling engine [8]:

- Story Actions and Events

  These are the actions that the user can do, multiple verbs make up events and are stored in flat data structures; an example from Crawford's engine is the trading sentence, "Subject Trades X (to) Direct Object (in return for) Y" [8]

- Drama and Story Manager

  This is the main algorithmic game/story engine that links everything together. It updates the game world, the characters and their goals and evaluates events and their impact on the plot. It also remembers past events and learns from their results.

- Personality and Emotional Models (Character Engine)

  These help the characters decide what decisions to make depending on certain character traits represented as dynamic numerical values that can change depending on interactions with other characters; for example an anger value may increase if insulted.

- Roles and Sequencing Engine

  The sequencing engine's job is to calculate what should happen next after an event. It makes sure the options available to a player are relevant to the previous event and that characters choices fit their role correctly. Some of this will be moved into the Character Engine for our design.

- History Database and Blackboard Systems

  History modules record events that characters do that could influence their future decisions and blackboard systems allow characters to share experiences and knowledge from their own point of view.

- Anticipation Engine

  This system allows characters to anticipate another's reaction to their behaviour and adjust their actions accordingly.

- Development Environment IDE

  These are the tools where story builders will specify, design, implement and evaluate their game stories. The development environment IDE must be comprehensive and easy to use.

Figure 1 gives an overview of the DISE architecture, which will sit on top of the Java Monkey Engine.

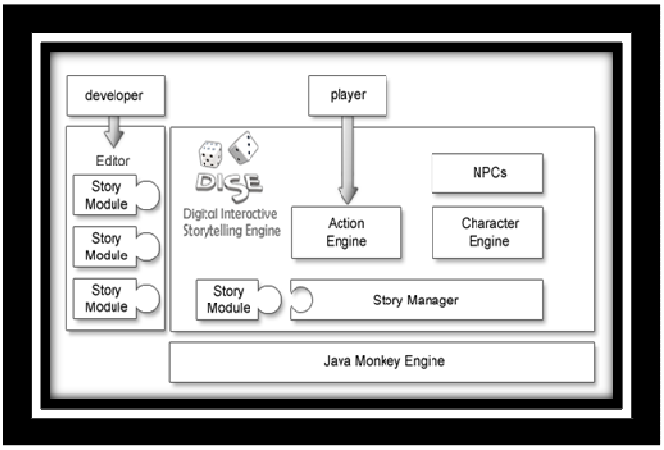

Figure 1

3.2 Action Engine

To interact with the story a player needs to be able to describe the actions that they wish to carry out. Players are given a choice of words that link together to form coherent sentences. There are a limited number of words, which are made available depending on the particular scene and context of the story at a given time.

The story developer can choose these verbs/actions by listing them in the scene verb XML file, which allows them to control to some extent the direction of the story. For example in a fairytale story for children the actions shoot/kill may not be appropriate and therefore can be omitted entirely from every scene.

The first word chosen is probably the most important and describes the type of action using a verb. Depending on which verb is chosen a number of other words will be needed to describe the rest of the action. These following words can be one of the following types:

- Noun – Object
- Noun – Character
- Extra Verb (mainly used when asking/telling another character to perform an action, or showing another character an action).
- Child objects of a previous object or character (e.g. Bob's arm is a child object of Bob).

Figure 2 shows the process used to narrow down the actions a player can carryout at certain points in the story, depending on the context of the current scene.

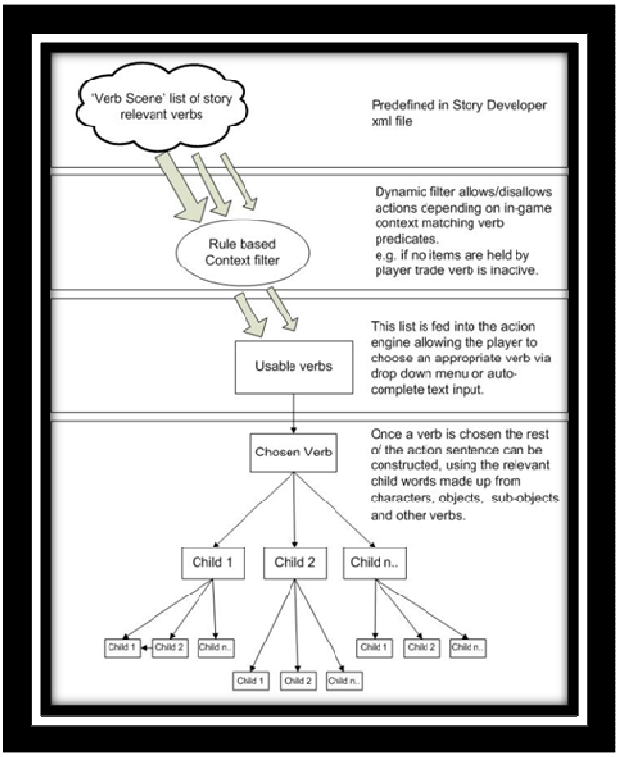

Figure 2

3.3 Character Engine

The character engine's job will be to maintain and update the actions and behaviour of each computer controlled non-player-character (NPC). For a character driven story to emerge each character has to behave differently to the others and make certain choices that progress the story in an interesting way as an individual. The character engine we propose is loosely based on the guidelines Crawford explains in his book [8].

3.3.1 Personality Model

To represent a unique personality, a personality model will be needed for each NPC [8]. The personality model consists of a range of variables:

- First person variables – these are represented by one dimensional arrays of characters and their traits. (E.g. `Honesty [character1] = 1`).
- Second person variables – two dimensional arrays of character relationships, with a subject and a focus character. (E.g. `Trust [character1, character2] = 1`).
- Third person variables – these more complicated interactions can be stored in three dimensional arrays that could describe what one person thinks a second person thinks about a third person. (E.g. `3rdPersonTrust [character1, character2, character3] = 1`).

These variables will be used in conjunction with the choice formulae to allow the NPCs to make appropriate decisions and carryout suitable actions depending on their mood, perception of others and personality traits/characteristics in a given situation.

3.3.2 Character Roles

Each verb/action contains a list of roles in their verb definition that describes which computer controlled characters should react to the event taking place and gives them a list of fitting reactions to choose from. For each different role there are four main elements to consider:

1. **Rules** to describe which character should react to the event and which role they fit into.
2. The **Modifiers** that change the personality variables of the character in the specific role, which reflects how they feel about the current action and how it physically affected them.
3. A list of **Reaction Verbs** to choose from and act out.
4. A **Choice Formula** that allows the character to choose the most suitable verb from the aforementioned list according to their updated personality model.

A role can be anything that a story builder defines in a verb and more roles could be added later if they are needed. Some actions have a focus character which can be easily defined as the character the action was performed on, other roles could be characters that passively observed the action and want to intervene.

3.3.3 Choice Formulae

The choice formulae for each verb are stored inside their verb definition in the verb dictionary and are different for each verb and each character role. This difference allows a clear separation of roles meaning that a mother, for example, may act differently to a child in a given situation. The choice formulae result in a Boolean flag or probability percentage value, depending on the type of options and rules. The variables in the formulae are substituted with the numbers taken from an element in the personality model. For example if *player2* wanted to lend a bike from *player1* to ride to the shop, their decision to say 'yes' could be represented by the formulae:

```
If (Trust [Player1, Player2] > 0.8f)
```

If *player2's* role was girlfriend to *player1* the formula may be changed to represent this relationship:

```
If (Trust [Player1, Player2] > 0.8f or
Affection [Player1, Player2] > 0.8f)
```

A large number of different roles can be stored in each verb and it's up to the story developer to choose which will be suitable for their story and to add simple new rules for any new roles or verbs.

3.3.4 Sequencing Actions

Once an action has been chosen by a non-player character (NPC) it will be queued up in their reaction list until the time is appropriate for its execution. This wait time depends on how long it takes to start the action and also on its audience requirements, which are stored inside the verbs definition (Figure 3). For example in a classic murder mystery scene, to kill character 1, both character 1 and character 2 must be alone together with no witnesses.

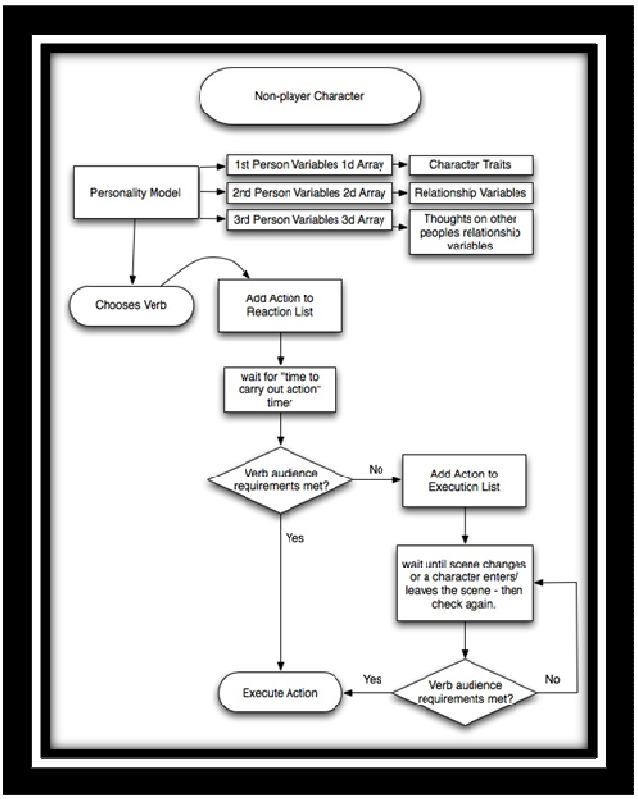

Figure 3

Once an action is performed other characters take turns to respond depending on their role (see 3.3.2 Character Roles). The responses are put into the following order:

- Fate's reaction – this is controlled by the story manager and can bend the reaction to fit the story developers outline.
- Witnesses' reaction – anyone present who isn't directly involved in the action but has a role.
- Focus characters' reaction – if the action has a main focus, e.g. in the action "punch Fred", Fred would be the focus character.
- Subject's reaction – the character that originally performed the action gets to go last and execute a follow-up action if necessary.

3.4 Story Manager

The story manager sits on top of the game engine and dynamically updates the state of the game world. The story manager completes three main goals:

- Monitor – the story's overall state needs to be monitored to make sure it always stays interesting and relevant. Some summary variables can give an impression of what has been happening and the general mood.
- Process – after monitoring the story, the data gathered must be processed to decide what needs to happen next.
- Change – the game world then has to be changed according to the processing results. This change will progress the story in the desired way.

The story manager will process data differently depending on the story module plug-in created by the developer, which includes the specific criteria that has to be met to change the story's path to follow their desired outline.

Crawford [8] lists the following techniques to evolve a story that can be carried out in the change stage:

- Environmental Manipulation
- Goal Injection
- Shifting Personalities
- Ticking Clock of Doom
- Dropping the Fourth Wall

3.5 Jess

Jess is a rule engine for the Java platform and a superset of the CLIPS programming language, developed by Ernest Friedman-Hill of Sandia National Labs. It was first written in late 1995. "Using Jess you can build Java software that has the capacity to 'reason' using knowledge you supply in the form of declarative rules. Jess is small, light, and one of the fastest rule engines available" [12]. It uses an enhanced version of the *Rete* algorithm to match the rules to the facts. It has features including backwards chaining and working memory queries, but mainly "can directly manipulate and reason about Java objects" [12], which will allow strong integration with the Java Monkey Engine we intend to use and cross platform compatibility execution via our Web Start deployment system [13].

3.6 Constraints Satisfaction Problem

Constraint Satisfaction Problems (CSPs) are used to solve many computer & mathematical problems [14]. They could be implemented in our engine to more efficiently filter out verbs in real time based on context and to check the choice formulae for the correct verb choices, forming the backbone of our AI and story engine.

The CCP (Clips Constraints Programming) module by Futtersack & Labat [15] allows Jess to solve CSPs and uses *The Seat Problem* as an example. Dynamic CSPs need a combination of heuristics and combinatorial search methods to be solved quickly.

CCP is composed of three modules [15]:

- The MAIN module contains the top level loop including the I/O rules.
- The SEARCH module implements a classical Chronological Backtrack algorithm.
- The PROPAGATION module implements the Forward Checking algorithm.

4. CONCLUSION

In this paper we have briefly reviewed some techniques that have been used previously to create stories with greater interactivity and provide players with a more immersive experience, where their actions cause real changes to themselves and the world around them. Using a combination of rule based languages, heuristic search methods and verb based actions; alongside a powerful multiplatform 3D games engine, we intend to create a game with a dynamic character driven storyline that doesn't bind players to a linear path and a reusable middleware codebase for story game developers.

5. REFERENCES

1. **Murray, Janet.** *First Person - From Game Story to Cyber Drama.* [ed.] Noah Wardrip-Fruin and Pat Harrigan. Cambridge : The MIT Press, 2004.

2. **Noyle, G.** Techniques of Written Storytelling Applied to Game Design. *Gamasutra.com.* [Online] 2006. [Cited: 12 February 2008.] http://www.gamasutra.com/features/20060426/noyle_01.shtml.

3. **Spector, W.** Next-Gen Storytelling Part One: What Makes a Story? *escapistmagazine.com.* [Online] 2007. [Cited: 12 February 2008.] http://www.escapistmagazine.com/news/view/70852-Next-Gen-Storytelling-Part-One-What-Makes-a-Story.

4. **IGDA.** Foundations of Interactive Storytelling. *IGDA.org.* [Online] 2001. [Cited: 12 february 2008.] http://www.igda.org/writing/InteractiveStorytelling.htm.

5. *Adaptative execution of game: unfolding a correct story.* **Champagnat, R, Estraillier, P and Prigent, A.** New York : ACM, 2006. Proceedings of the 2006 ACM SIGCHI international Conference on Advances in Computer Entertainment Technology (Hollywood, California, June 14 - 16, 2006). Vol. 266. ISBN 1-59593-380-8.

6. **Cavazza, M, Charles, F and Mead, S.** *Sex, Lies, and Video Games: an Interactive Storytelling Prototype.* Middlesbrough : University of Teesside CMS, 2002.

7. *Fuzzy cognitive goal net for interactive storytelling plot design.* **Cai, Yundong, Miao, Chunyan and Shen, Zhiqi.** New York : ACM, 2006. Proceedings of the 2006 ACM SIGCHI international Conference on Advances in Computer Entertainment Technology. (Hollywood, California, June 14 - 16, 2006). Vol. 266. ISBN 1-59593-380-8.

8. **Crawford, C.** *Chris Crawford on Interactive Storytelling.* Berkeley : New Riders, 2004.

9. **Zhiqi, S, et al.** *Goal-oriented Modelling for Intelligent Software Agents.* Singapore : Nanyang Technological University, 2005.

10. *Façade: An Experiment in Building a Fully-Realised Interactive Drama.* **Mateas, M. and Stern, A.** Atlanta : Literature, Communication and Culture and College of Computing, Georgia Tech., 2003.

11. **Crawford, C and Mixon, L. J.** Storytron Interactive Storytelling. *http://www.storytron.com/.* [Online] Storytron, Inc, 2008. [Cited: 1 October 2008.] http://www.storytron.com/.

12. Jess Home. *jessrules.com* . [Online] Sandia National Laboratories, 6 August 2008. [Cited: 7 October 2008.] http://herzberg.ca.sandia.gov/.

13. **CMP Research Students.** Web Starts. *Homura Game Development IDE.* [Online] JMU. [Cited: 23 September 2008.] http://java.cms.livjm.ac.uk/homura/downloads.php.

14. **Tsang, E.P.K.** *Foundations of Constraint Satisfaction.* London and San Diego : Academic Press, 1993. ISBN 0-12-701610-4.

15. **Futtersack, M and Labac, J. M.** CCP Constraint Programming within CLIPS. *Paris Descartes University.* [Online] Paris Descartes University. [Cited: 4 April 2008.] http://www.droit.univ-paris5.fr/futtersack/english/research/CCP/index.html.