Thompson, M., Padget, J. and Battle, S. (2015) 'An interactive, generative Punch and Judy show using institutions, ASP and emotional agents', in Dignum, V., Noriega, P., Sensoy, M. and Sichman, J., eds. *Coordination, Organizations, Institutions, and Norms in Agent Systems XI: COIN 2015. Lecture notes in computer science (9628).* **Cham: Springer, pp. 396-417.**

Official URL: http://dx.doi.org/10.1007/978-3-319-42691-4_22

ResearchSPAce

http://researchspace.bathspa.ac.uk/

An interactive, generative Punch and Judy show using institutions, ASP and emotional agents[*]

Matt Thompson[1][**], Julian Padget[1], and Steve Battle[2]

[1] Department of Computer Science,
University of Bath, UK
{m.r.thompson,j.a.padget}@bath.ac.uk
[2] Dept. Computer Science and Creative Technologies, University of the West of England, UK
steve.battle@uwe.ac.uk

Abstract. Using Punch and Judy as a story domain, we describe an interactive puppet show, where the flow and content of the story can be influenced by the actions of the audience. As the puppet show is acted out, the audience reacts to events by cheering or booing the characters. This affects the agents' emotional state, potentially causing them to change their actions, altering the course of the narrative. An institutional normative model is used to constrain the narrative so that it remains consistent with the Punch and Judy canon. Through this vignette of a socio-technical system (STS), comprising human and software actors, an institutional model – derived from narrative theory – and (simplistic) technological interaction artifacts, we begin to be able to explore some of the issues that can arise in STS through the prism of the World-Institution-Technology (WIT) model.

1 Introduction

Agent-based approaches for interactive narrative generation use intelligent agents to model the characters in a story. The agents respond to the interactions of a player with dialogue or actions fitting the shape of a story. However, these agents have little autonomy in their actions, bound as they are to the strict requirements of their role in the narrative.

Other approaches to balancing authorial control with player or character agency include the use of director agents [15], reincorporation of player actions back into the narrative [29] and mediation to prevent narrative-breaking actions [26].

An institutional model can be used as a normative framework for governing the actions of agents in a story. By describing the rules of a narrative in terms of social expectations, the agents are encouraged to perform certain types of actions while still retaining the option to break free of these expectations. As in society in the real world, breaking agreed norms comes with consequences, and only generally happens in exceptional circumstances. One situation where this may reasonably occur is when agents experience emotions. An agent experiencing an extreme emotion – in respect of some

---

emotional model – such as rage or depression, may be allowed to act unusually or uncharacteristically. Allowing characters to break with the norms of the narrative enables them to be 'pushed too far' by circumstances, with results that can add an extra dimension of richness to the telling.

There are two novel aspects to the approach we describe here: (i) the use of an institutional model to describe a narrative 'world' or domain, and (ii) how emotional models can give intelligent agents an alternative form of autonomy – from being limited by knowledge, reasoning capacity and time (bounded-rationality) and self-interest – both to act in idiosyncratic ways and to react emotionally to input from the audience.

Here we present an implementation in the form of an interactive Punch and Judy puppet show, in which the course of the story changes in accordance with the responses that come from the audience.

The structure of the system takes cues from the WIT model [22] which offers a pattern for analysing socio-technical systems through the interaction of three views of a system: (i) the world view (W), as human and software agents see it: in this case, the audience (human) and the actors (BDI agents) playing roles in Punch and Judy, (ii) the institutional view (I) that sets out the regulation of the system: in this case the narrative structure corresponding to Punch and Judy, captured in terms of Propp's [23] story moves and roles, and (iii) the technological view (T) that identifies the components (software and hardware) that enable the realization of the system, in this case, the means to capture audience input and the visualisation of the performance. More importantly, WIT emphasises the role of the institution both as regulator and monitor of behaviour, which is exactly what we see in our system, since through permission and obligation it directs the actors towards the conclusion of the narrative, while also observing their actions for adherence to the narrative structure.

The puppets in the show are each realised by belief-desire-intention (BDI) agents augmented with a valence, arousal, dominance (VAD) emotional model, which we describe in section 4. The story is modelled by a set of institutional norms (section 6) that describe the Punch and Judy story domain in terms of Propp's 'story moves' [23] (section 2). In section 6, we discuss the architecture and the means for the audience to interact with the system. The focus here is on the more technical aspects of the system and how the various components fit together, while more detail on the narrative side appears in [28].

2 Propp's story moves and roles

To express story events as an institution, we need some sort of formalisation for the analysis of the story – rather than an arbitrary selection of features – and so we look to narrative theory for inspiration. Instead of describing parts of the Punch and Judy story explicitly (such as 'Punch is expected to hit the policeman in this scene'), it is desirable to describe scenes in a more abstract way using roles ('The villain fights the victim in this scene'). The use of more general story fragments allows us to reuse them in multiple scenes, or even in other stories.

Narratology, and structuralism in particular, supply such generalised building blocks for stories. Russian formalism is an early movement in narrative theory that sought to

formalise the elements of narrative, and Vladimir Propp was a prominent figure in this school. One outcome of this movement was Propp's 1928 formalism derived from the study of Russian folktales, *The Morphology of the Folktale* [23], which is what we use to build a model to direct the course of the narrative. Propp is widely used – and criticised – in the domains of computational models of narrative and digital story telling [8], but retains appeal through it's simple but relatively effective modelling capacity. In his formalism, Propp identifies recurring characters, which become roles and motifs, which become action fragments, in Russian folklore, distilling them down to a concise syntax with which to describe stories. Propp's event-driven style translates comfortably to an institutional framework for event-based norms. However, while these action fragments fit the Punch and Judy story adequately, we note that the role labels can sound rather awkward because of the apparent semantic import of the textual label.

In Propp's formalism, characters have *roles*, such as *hero*, *villain*, *dispatcher*, *false hero*, and more. Characters performing a certain role are able to perform a subset of *story moves*, which are actions that make the narrative progress. For example, the *dispatcher* might send the *hero* on a quest, or the *victim* may issue an *interdiction* to the *villain*, which is then *violated*.

Propp defines a total of 31 distinct story functions. Each such function is denoted by a number and symbol in order to provide a succinct way of describing entire stories by reference to the constituent story functions. Examples of such functions are:

– One of the members of a family absents himself from home: *absentation*.
– An interdiction is addressed to the hero: *interdiction*.
– The victim submits to deception and thereby unwittingly helps his enemy: *complicity*.
– The villain causes harm or injury to a member of the family: *villainy*.

Each of these functions can vary in subtle ways. For example, the *villainy* function can be realised as one of 19 distinct forms of villainous deed, including *the villain abducts a person*, *the villain seizes the daylight*, and *the villain makes a threat of cannibalism*. These functions are enacted by characters following certain roles. Each role (or *dramatis persona* in Propp's definition) has a *sphere of action* consisting of the functions that they are able to perform at a particular point in the story. Propp defines seven roles each of which has distinct spheres of action: *villain*, *donor*, *helper*, *princess*, *dispatcher*, *hero*, and *false hero*. In a typical story, one story function will follow another as the tale progresses in a sequence of cause and effect. Propp's formalism does however also allow for simultaneous story functions.

2.1 Propp example: sausages and crocodile scene

To provide some context for Punch and Judy, since it is a peculiarly British phenomenon, although with Italian origins, we quote from Wikipedia:

> Punch and Judy is a traditional, popular, and usually very violent puppet show featuring Mr Punch and his wife, Judy. The performance consists of a sequence of short scenes, each depicting an interaction between two characters, most typically Mr Punch and one other character (who usually falls victim to Mr. Punch's club). It is often associated with traditional British seaside culture. The Punch and Judy show has roots in the 16th-century

Italian commedia dell'arte.
`http://en.wikipedia.org/wiki/Punch_and_Judy`, retrieved 2015-05-06.

The common elements of Punch and Judy are easily described in terms of Propp's story functions. Here we pick one scene to use as an example: the scene where Punch battles a crocodile in order to safeguard some sausages. In this scene, Joey the clown (our narrator) asks Punch to guard the sausages. Once Joey has left the stage, a crocodile appears and eats the sausages. Punch fights with the crocodile, but it escapes. Joey then returns to find that his sausages are gone. The corresponding story functions are:

1. Joey tells Punch to look after the sausages (*interdiction*).
2. Joey has some reservations, but decides to trust Punch (*complicity*).
3. Joey gives the sausages to Punch (*provision or receipt of a magical agent*).
4. Joey leaves the stage (*absentation*).
5. A crocodile enters the stage and eats the sausages (*violation*).
6. Punch fights with the crocodile (*struggle*).
7. Joey returns to find that the sausages are gone (*return*).

Some features of Punch and Judy map to story functions better than others (for example, it is debatable as to whether or not the sausages can be considered a "magical agent"), but for the most part Propp's formalism seems well suited to Punch and Judy. The advantage of using Propp to model the Punch and Judy story domain is that the story function concept captures the notion of actual story (brute) events counting as [11] story function (institutional) events and hence leading to the construction and evolution of institutional models.

3 Institutional model

An institution describes a set of 'social' norms describing the permitted and obliged behaviour of interacting agents. Noriega's 'Fish Market' thesis [3] describe how an institutional model can be used to regiment the actions of agents in a fish market auction. Several [2, 9, 5] extend this idea to build systems where institutions actively regulate the actions of agents, while still allowing them to decide what to do. We build on the work of Cliffe et al. [7] and Lee et al. [14] to adapt it for the world of narrative, using an institutional model to describe the story world of Punch and Judy in terms of Propp's story moves and character roles, through which the actors acquire powers and permissions appropriate to the character and the story function in which they are participating.

Institutional models use concepts from deontic logic to provide obligations and permissions that act on interacting agents in an environment. By combining this approach with Propp's concepts of *roles* and *story moves*, we describe a Propp-style formalism of Punch and Judy in terms of what agents are *obliged* and *permitted* to do at certain points in the story.

For example, in one Punch and Judy scene, a policeman enters the stage and attempts to apprehend Punch. According to the rules of the Punch and Judy world, Punch has an obligation to kill the policeman by the end of the scene (as this is what the audience

expects to happen, having seen other Punch and Judy shows). The policeman has an obligation to try his best to catch Punch. Both agents have permission to be on the stage during the scene. The policeman only has permission to chase Punch if he can see him (Punch is obliged to hide from him at the start of the scene).

The permissions an agent has, on the one hand, constrain the choices of actions available to them at any given moment. Obligations, on the other hand, affect the goals of an agent. Whether or not an agent actively tries to fulfil an obligation depends on their emotional state.

3.1 Institution example

To illustrate the application of institutional modelling, we here continue the 'sausages and crocodile' scene example from section 2.1, taking the Propp story functions and describing them in an institutional model. We define our institution in terms of *fluents*, *events*, *powers*, *permissions* and *obligations*, following [7], to which the interested reader is referred for the full details of the formal model, including the generate ($\mathcal{G}$) and consequence ($\mathcal{C}$) relations, which are only described here in sufficient depth for the model being presented.

Fluents These are properties that may or may not hold true at some instant in time, and that change over the course of time. *Institutional events* are able to *initiate* or *terminate* fluents at points in time. A fluent could describe whether a character is currently on stage, the scene of the story that is currently being acted out, or whether or not the character is happy at that moment in time. Domain fluents ($\mathcal{D}$) describe domain-specific properties that can hold at a certain point in time. In the Punch and Judy domain, these can be whether or not an agent is on stage, or their role in the narrative:

$$\mathcal{D} = \{\texttt{onstage}, \texttt{hero}, \texttt{villain}, \texttt{victim}, \texttt{donor}, \texttt{item}\}$$

Institutional fluents consist of (institutional) *powers*, *permissions* and *obligations*. An **institutional power** ($\mathcal{W}$) describes whether or not an external event has the authority to generate a meaningful institutional event. Taking an example from Propp's formalism, an *absentation* event can only be generated by an external event brought about by a *donor* character (such as their leaving the stage). Therefore, any characters other than the donor character would not have the institutional power to generate an *absentation* institutional event when they leave the stage. The possible empowerments (institutional events) from Propp used in Punch and Judy are:

$$\mathcal{W} = \{\texttt{pow(introduction)}, \texttt{pow(interdiction)}, \texttt{pow(give)},$$
$$\texttt{pow(absentation)}, \texttt{pow(violation)}, \texttt{pow(return)}\}$$

Permissions ($\mathcal{P}$) are associated with external actions that agents are permitted to do at a certain instant in time. These can be thought of as the set of *socially permitted* actions available to an agent. While it is possible for an agent to perform other actions, societal norms usually discourage them from doing so. For example, it would not make sense in

the world of Punch and Judy if Punch were to give the sausages to the Policeman. It is always Joey who gives the sausages to Punch. Also, it would be strange if Joey were to do this in the middle of a scene where Punch and Judy are arguing. We make sure agents' actions are governed so as to allow them only a certain subset of permitted actions at any one time. The set of permission fluents is:

$$\mathcal{P} = \{\texttt{perm(leavestage)}, \texttt{perm(enterstage)}, \texttt{perm(die)}, \texttt{perm(kill)},$$
$$\texttt{perm(hit)}, \texttt{perm(give)}, \texttt{perm(fight)}\}$$

Obligations ($\mathcal{O}$) are institutional facts that contain actions agents *should* do before a certain deadline. If the action is not performed in time, a *violation event* is triggered, which may result in a penalty being incurred. While an agent may be obliged to perform an action, it is entirely their choice whether or not they actually do so. They must weigh up whether or not pursuing other courses of action is worth accepting the penalty that an unfulfilled obligation brings.

Anybody who has seen a Punch and Judy show knows that at some point Joey tells Punch to guard some sausages, before disappearing offstage. Joey's departure is modelled in the institution as the *absentation* event. It could also be said that Joey has an obligation to leave the stage as part of the *absentation* event, otherwise the story function is violated. This can be described in the institution as:

$$\mathcal{O} = \{\texttt{obl(leavestage}, \texttt{absentation}, \texttt{viol(absentation))}\}$$

The first argument is the external event that must be triggered according to the obligation, the second argument is the institutional deadline event, and the third argument is the violation event which is triggered if the obligation is not fulfilled before the deadline.

Events Cliffe's model specifies three types of **event**: *external events* (or 'observed events', $\mathcal{E}_{obs}$), *institutional events* ($\mathcal{E}_{instevent}$) and *violation events* ($\mathcal{E}_{viol}$). Examples of each are given in Figure 1. *External events* are observed to happen in the agents' environment, which can *generate institutional events* which occur only within the institional model, leading to the *initiation* or *termination* of (domain) fluents, permissions, obligations or institutional powers. An external event could be an agent leaving the stage, an agent hitting another, or an agent dying. Internal events include narrative events such as scene changes, or the triggering of Propp story functions such as *absentation* or *interdiction* (described in section 2). *Violation* is the name of a Propp story function, and is included as an internal event, although it has no relation to the violation events of an institution. Violation events occur when an agent has failed to fulfil an obligation before the specified deadline. These can be implemented in the form of a penalty, by decreasing an agent's health, for example.

Event Generation and Consequences An **event generation** function, $\mathcal{G}$, describes how events ($\mathcal{E}$, usually external, but can also be internal) can generate other (usually institutional) events, conditional upon the current institutional state ($\mathcal{X}$). This is the counts-as relation. For example, if an agent leaves the stage while the *interdiction* event

$$\begin{aligned}
\mathcal{E}_{obs} = \{&\texttt{startshow}, \texttt{leavestage}, \texttt{enterstage}, \texttt{die}, \texttt{give}, \\
&\texttt{harmed}, \texttt{hit}, \texttt{fight}, \texttt{kill}, \texttt{escape}\} \quad (1) \\
\mathcal{E}_{instevent} = \{&\texttt{introduction}, \texttt{interdiction}, \texttt{receipt}, \texttt{absentation}, \\
&\texttt{violation}, \texttt{return}, \texttt{struggle}, \texttt{defeat}, \texttt{complicity}, \\
&\texttt{victory}, \texttt{escape}\} \quad (2) \\
\mathcal{E}_{viol} = \{&\texttt{viol(introduction)}, \texttt{viol(interdiction)}, \texttt{viol(receipt)}, \\
&\texttt{viol(absentation)}, \texttt{viol(violation)}, \texttt{viol(return)}, \\
&\texttt{viol(struggle)}, \texttt{viol(defeat)}, \texttt{viol(complicity)} \\
&\texttt{viol(victory)}, \texttt{viol(escape)}\} \quad (3)
\end{aligned}$$

Fig. 1. External, institutional and violation events for Punch and Judy

holds, they trigger the *leavestage* event. This combination generates the *absentation* institutional event (rule 7). Further examples appear in figure 2.

Event generation functions follow a $\langle\texttt{preconditions}\rangle \rightarrow \{\texttt{postconditions}\}$ format. The preconditions consist of a set of fluents that hold at that time, along with an event to have occurred. The postconditions are the events that are generated. The generation functions are used to generate internal, institutional events from external events.

Consider the Punch and Judy scenario described in section 2.1. There are seven institutional events (story functions) that occur during this scene: *interdiction*, *complicity*, *receipt* (from Propp's *receipt of a magical agent*) *absentation*, *violation*, *struggle*, *return*. These institutional events are all generated by external events. The *interdiction* is generated when Joey tells Punch to protect the sausages. Punch agreeing amounts to *complicity*. Joey *gives* punch the sausages (*receipt*), then leaves the stage (*absentation*). The crocodile eating the sausages is a *violation* of Punch's oath, the agents fight (*struggle*), then Joey enters the stage again (*return*).

It is desirable that these story functions occur in this sequence in order for a satisfying narrative to emerge. Agents may decide to perform actions that diverge from this set of events, but the institution is guiding them towards the most fitting outcome for a *Punch and Judy* world. For this reason, a currently active story function can be the precondition for event generation. For example, the *receipt* event may only be triggered if an agent externally performs a *give* action **and** if the *complicity* event currently holds (rule 6). Examples of event generation function for this scenario, complete with preconditions, are listed in rules 4–10 (Figure 2).

Consequences consist of fluents, permissions and obligations that are *initiated* ($\mathcal{C}^{\uparrow}$) or *terminated* ($\mathcal{C}^{\downarrow}$) by institutional events. For example, the institutional event *receipt* initiates the donor agent's permission to leave the stage, triggering the *absentation* event (rule 12). When the *interdiction* event is currently active and a *violation* event occurs, the interdiction event is terminated (19). Rules 11–20 in Figures 3 and 4 describe the initiation and termination of fluents in the Punch and Judy sausages scene detailed in section 2.1.

$$\mathcal{G}(\mathcal{X},\mathcal{E}): \begin{cases} \langle \emptyset, telprotect(\texttt{donor}, \texttt{villain}, \texttt{item})\rangle \rightarrow \{interdiction\} & (4) \\ \langle \{interdiction\}, agree(\texttt{villain}))\rangle \rightarrow \{complicity\} & (5) \\ \langle \emptyset, give(\texttt{donor}, \texttt{villain}, \texttt{item}))\rangle \rightarrow \{receipt\} & (6) \\ \langle \{interdiction\}, leavestage(\texttt{donor})\rangle \rightarrow \{absentation\} & (7) \\ \langle \{interdiction\}, harmed(\texttt{item})\rangle \rightarrow \{violation\} & (8) \\ \langle \{interdiction, absentation\}, enterstage(\texttt{donor})\rangle \rightarrow \{return\} & (9) \\ \langle \emptyset, hit(\texttt{donor}, \texttt{villain})\rangle \rightarrow \{struggle\} & (10) \end{cases}$$

Fig. 2. Event generation in the sausage scene

$$\mathcal{C}^{\uparrow}(\mathcal{X},\mathcal{E}): \begin{cases} \langle \emptyset, \texttt{interdiction}\rangle \rightarrow \{\text{active}(\texttt{interdiction}), \\ \qquad\qquad\qquad \text{perm}(\texttt{give}(\texttt{donor}, \texttt{villain}, \texttt{item}))\} & (11) \\ \langle \emptyset, \texttt{receipt}\rangle \rightarrow \{\text{perm}(\texttt{leavestage}(\texttt{donor}))\} & (12) \\ \langle \{active(absentation)\}, \\ \texttt{enterstage}(\texttt{villain})\rangle \rightarrow \{\text{obl}(\texttt{eat}(\texttt{villain}, \texttt{sausages}), \\ \qquad\qquad\qquad \texttt{return}, \text{viol}(violation))\} & (13) \\ \langle \{active(interdiction)\}, \\ \quad \texttt{leavestage}(\texttt{donor})\rangle \rightarrow \{\text{obl}(\texttt{enterstage}(\texttt{donor}), \\ \qquad\qquad\qquad \texttt{eat}(\texttt{villain}, \texttt{sausages}), \\ \qquad\qquad\qquad \text{viol}(return))\} & (14) \\ \{active(interdiction)\}, \\ \qquad\qquad \texttt{violation}\rangle \rightarrow \{\text{perm}(\texttt{enterstage}(\texttt{dispatcher}))\} & (15) \\ \langle \{active(absentation), \\ \quad active(violation)\}, \\ \qquad\qquad \texttt{return}\rangle \rightarrow \{\text{perm}(\texttt{hit}(\texttt{donor}, \texttt{villain}))\} & (16) \end{cases}$$

Fig. 3. Fluent initiation in the sausage scene

$$\mathcal{C}^{\downarrow}(\mathcal{X},\mathcal{E}): \begin{cases} \langle \emptyset, \texttt{interdiction}\rangle \rightarrow \{\text{perm}(\texttt{give}(\texttt{donor}, \texttt{villain}, \texttt{item}))\} & (17) \\ \langle \{active(interdiction)\}, \\ \qquad\qquad \texttt{absentation}\rangle \rightarrow \{\text{perm}(\texttt{leavestage}(\texttt{donor}))\} & (18) \\ \langle \{active(interdiction)\}, \\ \qquad\qquad \texttt{violation}\rangle \rightarrow \{active(interdiction)\} & (19) \\ \langle \{active(absentation), \\ \quad active(violation)\}, \\ \qquad\qquad \texttt{return}\rangle \rightarrow \{active(absentation)\} & (20) \end{cases}$$

Fig. 4. Fluent termination in the sausage scene

4 VAD emotional model

In order to make the agents acting out the Punch and Judy show more believable, we apply an emotional model to affect their actions and decisions. For this, we use the valence-arousal (circumplex) model first described by Russell [27]. To give each character its own distinct personality, we extend this model with an extra dimension: dominance, as used by Ahn et al. [1] in their model for conversational virtual humans. This dominance level is affected by the reactions of the audience to the agents' actions. For example, Judy may become more dominant as her suggestions to hit Punch with a

Arousal

Furious	Vigilant	Excited

High Dominance — Valence

Angry	Alert	Malicious

Sulky	Annoyed	Vicious

Embarrased	Surprised	Delighted

Medium Dominance — Valence

Unhappy	Neutral	Happy

Anxious	Sleepy	Glad

Afraid	Scared	Empathic

Low Dominance — Valence

Depressed	Doubtful	Compassionate

Bored/sad	Tired	Peaceful

The VAD model illustrates how valence, arousal and dominance values map to identifiable emotions. Valence, arousal and dominance can each have a value of low, medium or high. This allows the agents to have a total of 27 distinct emotional states. The valence and arousal levels of each agent are affected by the actions of other agents. For example, a character being chased around the stage by Punch will see their valence level drop while their arousal increases. According to Russell's circumplex model of emotion [27], this would result in them becoming *afraid* (if their dominance level is low).

An agent's emotional state affects its ability to fulfil its institutional obligations. An agent that is *furious* might have no problem carrying out an obligation that requires them to kill another agent. If that same agent is *happy* or *depressed*, however, they might not have the appropriate motivation to perform such a violent action.

Fig. 5. VAD emotional values (figure adapted from Ahn et al.) [1]

stick are cheered on by the audience, emboldening her into acting out her impulses. A detailed description appears in the text in Figure 5.

It is important to note that the emotional model is part of the agent belief state, and not held in the institution. We want to explore how the characters of the story might be able to choose actions based on their emotional state. While the institution could theoretically calculate the emotional state for each agent in turn and dictate this to them along with the norms of the narrative, it makes sense to decouple this feature from the narrative institution in order to separate the characters from the events of the story.

Agents' emotional states change according to their interactions with the audience. This is unrelated to what is happening in the narrative, and so this underscores the decision not to include any emotional modelling in the institution. Also, we want the agents to have some degree of freedom within the narrative world. They should be allowed to determine their emotions themselves, so that in extreme emotional states

Listing 1.1. Emotional rules for a character with medium dominance

```
1  emotion(sleepy) :- valence(0) & arousal(-1) & dominance(0).
2  emotion(neutral) :- valence(0) & arousal(0) & dominance(0).
3  emotion(surprised) :- valence(0) & arousal(1) & dominance(0).
4  emotion(anxious) :- valence(-1) & arousal(-1) & dominance(0).
5  emotion(unhappy) :- valence(-1) & arousal(0) & dominance(0).
6  emotion(embarrassed) :- valence(-1) & arousal(1) & dominance(0).
7  emotion(glad) :- valence(1) & arousal(-1) & dominance(0).
8  emotion(happy) :- valence(1) & arousal(0) & dominance(0).
9  emotion(delighted) :- valence(1) & arousal(1) & dominance(0).
```

they can perform 'irrational' or 'extreme' actions that may not necessarily fit into the narrative.

4.1 VAD emotions in Jason

Emotions are implemented as beliefs inside an agent. An agent believes it has a certain level of valence, arousal and dominance, and it works out its emotional state based on a combination of these three factors. When the audience cheers or boos them, this changes the belief holding the relevant emotional variable, and their emotional state as a whole is recalculated.

Valence, arousal and dominance values can take values of -1 (low), 0 (medium) or 1 (high). Listing 1.1 shows the emotional belief rules for an agent with medium dominance (a dominance level of 0). Note that an agent maintains beliefs about both its current emotion label (such as sleepy or happy) and the separate valence, arousal and dominance values at the same time. Similar sets of rules handle the belief emotion for the other dominance levels.

Every time an emotional variable (valence, arousal, or dominance) changes, an agent's emotion is changed according to the rules in listing 1.1. While an agent's valence, arousal and dominance belief values affect the way it makes decisions internally, the results of combinations of these values (sleepy, happy, etc) are broadcast as external actions. The reason for this is that an agent's emotional state may affect the way in which the character is animated: changing the speed at which they move or turning their smile into a frown, for example. For this reason, whenever an emotional change takes place, the new emotion is published as an external action of the agent so that observing entities may perceive it. The Bath sensor framework described in section 6 provides the means for this evidence of the agent's internal state change to be received by the animation system and reflected accordingly in the display.

Listing 1.2 shows the AgentSpeak rules describing how an agent's valence and dominance levels are changed by the audience cheering or booing their actions. These AgentSpeak plans describe what the agent should do in response to a goal addition (denoted by a +! at the start of the plan name) or a belief addition (prefixed by a simple +). In the case of Listing 1.2, the +!changeMood plan updates the agent's emotional state based on its valence-arousal-dominance values and broadcasts the result as an external action. The +response plans raise or lower an agent's valence and dominance levels depending on whether the agent perceives a "boo" or "cheer" response from the audience.

Listing 1.2. AgentSpeak rules for changing an agent's emotional values from audience responses

```
1  +!changeMood
2    <- ?emotion(Z);
3        emotion(Z).
4  +response(_, boo) : asking
5    <- -+valence(-1);
6        -+dominance(-1);
7        !changeMood.
8  +response(_, cheer) : asking
9    <- -+valence(1);
10       -+dominance(1);
11       !changeMood.
```

An agent announces what they intend to do, then waits three seconds. During this time, they have the belief that they are 'asking' the audience, and listen for a response. A boo reduces an agent's valence and dominance, while a cheer raises them. For each response, the `changeMood` goal is triggered, which looks up and broadcasts the agent's emotional state to the other agents and environment.

5   Agent decision making

The agents choose which goals to pursue according to three factors: their permitted actions, their obliged actions and their emotional state. Though obliged actions are given priority, and while agents' decisions are generally constrained by their permitted actions, an agent's emotional state has the final say in its decisions. In this way, an agent will follow the social norms of the narrative, but only according to their own mood.

Agent goals and plans The agents are implemented using a belief-desire-intention (BDI) psychological model using the Jason platform [4]. An agent's knowledge about the state of their world and themselves are stored as *beliefs*, with new information coming in from the environment getting added to their belief base as *percepts*, which are ephemeral and only last for one reasoning cycle of an agent.

Agents are created with goals and plan libraries. Any goal that an agent is set on carrying out at any point is an *intention*, whereas a goal that an agent has but is not yet pursuing is a *desire*. Plan libraries describe the steps agents need to take in order to achieve goals, as well as how to react to changes in agents' environments.

Norms as percepts When an event occurs, it is added to the event timeline, which is used to query the ASP (Answer Set Programming) solver to obtain the set of norms that hold after the new event has occurred. The new permissions and obligations are then added to each agent as *percepts*. Each time this happens, the set of permitted and obliged actions that an agent sees is changed to be only those that apply at that instant in time, with the previous norms being discarded.

Agents choose between permitted and obliged actions based on their emotional state at the point of decision making. Obliged actions are given a higher priority over permitted ones for most of the emotional states that an agent can be in, though not always. If an

agent is in a sulky mood, for example, they may decide to ignore what they are obliged to do by the narrative, even though they know there will be consequences.

For example, in the scene where Joey gives the sausages to Punch, Punch may see that he has permission to eat the sausages, drop them, fight the crocodile, run away (leave the stage) or shout for help at the crocodile or audience. His obligation for the scene, in accordance with the Punch and Judy narrative world, is to either eat the sausages himself, or let the crocodile have them. This ends Propp's *interdiction* story function with a *violation* function. Note that his obligation in this case is not to guard the sausages as asked to by Joey. While Joey's entrusting of the sausages is certainly an obligation in itself, Punch's main obligations are to the narrative. Lesser obligations towards characters in the story can be implemented as having a lower prority than those of the story itself.

Similarly, at times of extreme emotion, an agent may decide to disregard their set of permitted actions entirely, instead acting out their innermost desires. For example, an angry Punch might decide to just attack Joey instead of agreeing to look after the sausages, or he might just decide to give up and leave if he is depressed. The key point is that the norms act as the will of the *narrative*, guiding the story forward, rather than a strict set of rules that the agents must follow at all costs.

Violation events add percepts to the agents telling them that they are in violation of the narrative norms. Once an agent receives such a percept, an emotional variable is changed. Typically, their dominance will decrease. The reasoning behind this is that if agents are unwilling to participate in the story, they should have less influence in its course of events.

6   Architecture

Multi-Agent System We use the JASON framework for belief-desire-intention (BDI) agents [4], programming our agents in the AgentSpeak language. The VAD emotional model is represented inside each agent as a set of beliefs. Each agent has beliefs for its *valence*, *arousal* and *dominance* levels, each of which can take the value of low, medium or high, as discussed in section 4. This combination of VAD values creates one of the 27 emotional states shown in figure 5, affecting whether or not an agent breaks from its permitted or obliged behaviour.

Institutional Framework To describe our institutional model, we use InstAL [7], a domain-specific (action) language for describing institutions that compiles to AnsProlog, a declarative programming language for Answer Set Programming (ASP). InstAL's semantics are inspired by the Situation Calculus [24] and the Event Calculus [13]. InstAL describes how external events generate institutional events, which then initiate or terminate fluents that hold at certain points in time. These fluents can include the permissions and obligations that describe what an agent is permitted or obliged to do when, as described in section 3. For example, if an agent with the role of *dispatcher* leaves the stage, it generates the Propp *absentation* move in the institution, but only if the *interdiction* function is active (i.e., the *activeFunction(interdiction)* fluent holds):

```
1  leaveStage(X) generates intAbsentation(X)
2      if role(X, dispatcher), activeFunction(interdiction);
```

Listing 1.3. Encoding of Propp's *absentation* function

```
1  intAbsentation(X) initiates activeFunction(
       absentation);
2  intAbsentation(X) initiates perm(harm(Y, Z)) if
       role(Y, villain), objStage(Z), onStage(Y),
       activeFunction(interdiction);
3  intAbsentation(X) initiates perm(harm(Y, Z)) if
       role(Y, ambusher), objStage(Z), onStage(Y),
       activeFunction(interdiction);
4  intAbsentation(X) initiates perm(enterStage(X)),
       activeFunction(absentation);
5  intAbsentation(X) initiates perm(enterStage(croc
       ) if objStage(sausages);
6  intAbsentation(X) terminates onStage(X), perm(
       leaveStage(X));
```

The *activeFunction(absentation)* function holds after any *intAbsentation* institutional event, indicating that that Propp function is currently underway

The *absentation* function gives the villain permission to harm an object, if both are on stage and the *interdiction* function is active

Same as above, but for *ambusher* role

The absented character has permission the re-enter the stage at any point during the *absentation* function

The crocodile has permission to enter the stage if the sausages are on stage

The *absentation* function means that the absented character is no longer on stage, and cannot leave the stage (as they have already done so)

which generates the following AnsProlog code:

```
1  occurred(intAbsentation(X),pj,I) :-
2     occurred(leaveStage(X),pj,I),
3     holdsat(pow(pj,intAbsentation(X)),pj,I),
4     holdsat(role(X,donor),pj,I),
5     holdsat(activeFunction(interdiction),pj,I),
6     agent(X), inst(pj), instant(I).
```

The internal *absentation* event occurs if the following conditions are met:

• *X* leaves the stage

• *X* has the power to leave the stage

• *X* has the role of donor

• the *interdiction* function is active

• *X* is an agent, pj is an institution, *I* is an instant

The *absentation* institutional event gives the crocodile permission to enter the stage if there are any sausages on the stage. It also terminates the permission of the absented agent to leave the stage, as they have already done so:

```
1  intAbsentation(X) initiates perm(enterStage(croc
       ) if objStage(sausages);
2  intAbsentation(X) terminates onStage(X), perm(
       leaveStage(X));
```

The *absentation* function gives the crocodile permission to enter the stage if the sausages are on stage

The *absentation* function means that once X leaves the stage, they are no longer on stage

which generates the following:

```
1  initiated(perm(enterStage(croc)),pj,I) :-
2     occurred(intAbsentation(X),pj,I),
3     holdsat(live(pj),pj,I), inst(pj),
4     holdsat(objStage(sausages),pj,I),
5     agent(X), inst(pj), instant(I).
```

The crocodile gets permission to enter the stage if the following conditions are met:

• the *absentation* function event has occured

• the pj institution is running

• the sausages are on stage

By combining statements such as the above, we can build a complete description of the sausages scene in terms of agent norms, such as the Propp *absentation* function, shown in Listing 1.3. InstAL rules like those shown above and in Listing 1.3 are compiled into AnsProlog, then we use the *clingo* answer set solver [10] to ground the program, and 'solve' queries by finding all permissions and obligations that apply to any agents, given

Listing 1.4. Possible full trace for sausages scene

```
1   observed(startShow,pj,0).
2   observed(startScene(sausages),pj,1).
3   observed(enterStage(punch),pj,2).
4   observed(enterStage(joey),pj,3).
5   observed(say(joey, give),pj,4).
6   observed(say(joey,  protect),pj,5).
7   observed(give(joey,punch,sausages),pj,6).
8   observed(leaveStage(joey),pj,7).
9   observed(say(punch, harm),pj,8).
10  observed(enterStage(croc),pj,9).
11  observed(take(croc, sausages),pj,10).
12  observed(eat(croc, sausages),pj,11).
13  observed(leaveStage(croc),pj,12).
14  observed(enterStage(joey),pj,13).
15  observed(hit(joey, punch),pj,14).
16  observed(leaveStage(joey),pj,15).
17  observed(leaveStage(punch),pj,16).
18  observed(startScene(end),pj,17).
```

The show has started

The sausages scene has started

Punch has entered the stage

Joey has entered the stage

Joey says he will give Punch the sausages

Joey tells Punch to protect the sausages

Joey gives the sausages to Punch

Joey leaves the stage

Punch says he will eat the sausages

The crocodile enters the stage

The crocodile takes the sausages

The crocodile eats the sausages

The crocodile leaves the stage

Joey enters the stage

Joey hits Punch

Joey leaves the stage

Punch leaves the stage

The scene ends

a sequence of events as the query input. The agents' percepts are then updated with their permitted and obliged actions from that time instant onwards. Thus, the institutional model acts as a social narrative sensor, interpreting actors' actions in the context of the combination of the concrete narrative and the abstract story moves which detach (instantiate) the norms that guide the actors in the direction of the conclusion of the story arc.

A query is simply a list of external events in chronological order, also called a *trace*. A possible trace describing the actions of agents acting out the *sausages* is described in Listing 1.4. The 'pj' in the trace is the name of the institution that observes the events, while the number is the enumeration of events in the sequence.

Each *observed* event triggers a corresponding *occurs* event inside the institution, as determined by the generates relation. Listing 1.5 shows an extract from an answer set output for the trace queried against the ASP description of the sausages scenario, for events 5 to 7 of the scene. Starting with an initial set of fluents that hold at instant 5, only fluents that have been initiated and not terminated hold at the next instant. For ease of reading, the listing only shows roles that hold at certain instants when they have some effect on the scene, although in practice, all role fluents hold throughout the scene. Figure 6 shows a visualisation of the answer set for the trace in listing 1.4.

Bath Sensor Framework The components communicate using the Bath Sensor Framework (BSF) [14], through publish / subscribe-style communication between distributed software components, in this case connecting intelligent agents with their virtual environments. It currently uses the XMPP publish/subscribe protocol for communication between agents, environment and other software components. Each agent subscribes to

Listing 1.5. Sausages scene using ASP

```
1   holdsat(perm(say(donor, protect)),pj,5).
2   holdsat(role(joey, donor),pj,5).
3   occurred(say(joey, protect),pj,5).
4
5   initiated(activeFunction(interdiction),pj,6).
6   initiated(perm(give(donor, villain, object)),pj,6).
7   terminated(perm(say(donor, protect)),pj,6).
8   holdsat(perm(give(donor, villain, object)),pj,6).
9   holdsat(activeFunction(interdiction),pj,6).
10  holdsat(role(joey, donor),pj,6).
11  holdsat(role(punch, villain),pj,6).
12  holdsat(object(sausages),pj,6).
13  occurred(give(joey, punch, sausages),pj,6).
14
15  initiated(activeFunction(absentation),pj,7).
16  initiated(perm(leavestage(donor)),pj,7).
17  terminated(perm(give(donor, villain, object)),pj,7).
18  holdsat(activeFunction(interdiction),pj,7).
19  holdsat(activeFunction(absentation),pj,7).
20  holdsat(role(joey, donor),pj,7).
21  holdsat(perm(leavestage(donor)),pj,7).
```

Annotations (right margin):
- The donor role can tell someone to protect an object at instant 5
- Joey is the donor at instant 5
- Joey tells another character to protect an object at instant 5
- The *interdiction* function becomes active at instant 6
- The donor has permission to give an object to the villain at instant 6
- The donor can no longer tell anyone to protect an object at instant 6
- The donor can give an object to the villain at instant 6
- The *interdiction* function is active at instant 6
- Joey is the donor at instant 6
- Punch is the villain at instant 6
- Sausages are an object at instant 6
- Joey gives the sausages to Punch at instant 6
- The *absentation* function becomes active at instant 7
- The donor can leave the stage at instant 7
- The donor can give an object to the villain at instant 7
- The *interdiction* function is active at instant 7
- The *absentation* function is active at instant 7
- Joey is the donor at instant 7
- The donor can leave the stage at instant 7

receive notifications of environment changes via the appropriate topic node in the XMPP server, which relays messages between publishers and subscribers. If any environment change occurs, all subscribed agents are informed of the changes.

Audience Interaction The puppet show is designed to be run in front of either a single user's computer, or on a large display in front of an audience. The user/audience is instructed to cheer or boo the actions of the characters of the show, which will be picked up by a microphone and 'heard' by the agents. This will then affect the emotional state of the agents and change the actions they make in the show. Their actions are constrained by the set of 'Punch and Judy' world norms as described in the institutional model.

There are many different ways in which the audience's responses can affect the outcomes of the show. If the audience craves a more 'traditional' Punch and Judy experience, then they can cheer Punch into beating and killing all of his adversaries (including his wife, Judy). Alternatively, a more mischievous audience could goad Judy into killing Punch and then taking over his role as sadist and killer for the rest of the
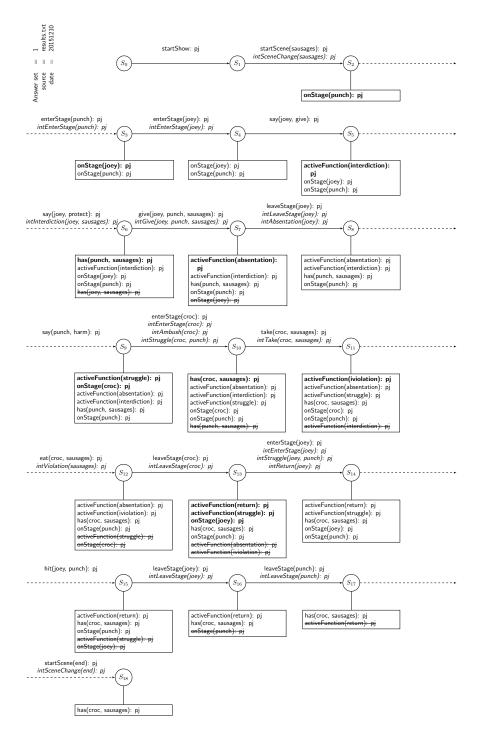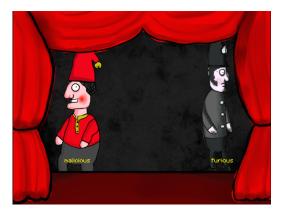
startShow: pj

startScene(sausages): pj
intSceneChange(sausages): pj

S_0 → S_1 → S_2

onStage(punch): pj

enterStage(punch): pj
intEnterStage(punch): pj

enterStage(joey): pj
intEnterStage(joey): pj

say(joey, give): pj

S_3 → S_4 → S_5

onStage(joey): pj
onStage(punch): pj

onStage(joey): pj
onStage(punch): pj

activeFunction(interdiction): pj
onStage(joey): pj
onStage(punch): pj

say(joey, protect): pj
intInterdiction(joey, sausages): pj

give(joey, punch, sausages): pj
intGive(joey, punch, sausages): pj

leaveStage(joey): pj
intLeaveStage(joey): pj
intAbsentation(joey): pj

S_6 → S_7 → S_8

has(punch, sausages): pj
activeFunction(interdiction): pj
onStage(joey): pj
onStage(punch): pj
~~has(joey, sausages): pj~~

activeFunction(absentation): pj
activeFunction(interdiction): pj
has(punch, sausages): pj
onStage(punch): pj
~~onStage(joey): pj~~

activeFunction(absentation): pj
activeFunction(interdiction): pj
has(punch, sausages): pj
onStage(punch): pj

say(punch, harm): pj

enterStage(croc): pj
intEnterStage(croc): pj
intAmbush(croc): pj
intStruggle(croc, punch): pj

take(croc, sausages): pj
intTake(croc, sausages): pj

S_9 → S_{10} → S_{11}

activeFunction(struggle): pj
onStage(croc): pj
activeFunction(absentation): pj
activeFunction(interdiction): pj
has(punch, sausages): pj
onStage(punch): pj

has(croc, sausages): pj
activeFunction(absentation): pj
activeFunction(interdiction): pj
activeFunction(struggle): pj
onStage(croc): pj
onStage(punch): pj
~~has(punch, sausages): pj~~

activeFunction(iviolation): pj
activeFunction(absentation): pj
activeFunction(struggle): pj
has(croc, sausages): pj
onStage(croc): pj
onStage(punch): pj
~~activeFunction(interdiction): pj~~

eat(croc, sausages): pj
intViolation(sausages): pj

leaveStage(croc): pj
intLeaveStage(croc): pj

enterStage(joey): pj
intEnterStage(joey): pj
intStruggle(joey, punch): pj
intReturn(joey): pj

S_{12} → S_{13} → S_{14}

activeFunction(absentation): pj
activeFunction(iviolation): pj
has(croc, sausages): pj
onStage(punch): pj
~~activeFunction(struggle): pj~~
~~onStage(croc): pj~~

activeFunction(return): pj
activeFunction(struggle): pj
onStage(joey): pj
has(croc, sausages): pj
onStage(punch): pj
~~activeFunction(absentation): pj~~
~~activeFunction(iviolation): pj~~

activeFunction(return): pj
activeFunction(struggle): pj
has(croc, sausages): pj
onStage(joey): pj
onStage(punch): pj

hit(joey, punch): pj

leaveStage(joey): pj
intLeaveStage(joey): pj

leaveStage(punch): pj
intLeaveStage(punch): pj

S_{15} → S_{16} → S_{17}

activeFunction(return): pj
has(croc, sausages): pj
onStage(punch): pj
~~activeFunction(struggle): pj~~
~~onStage(joey): pj~~

activeFunction(return): pj
has(croc, sausages): pj
~~onStage(punch): pj~~

has(croc, sausages): pj
~~activeFunction(return): pj~~

startScene(end): pj
intSceneChange(end): pj

S_{18}

has(croc, sausages): pj

Fig. 6. Trace visualisation

The animation engine that shows the visual output of the agents actions is written in Javascript and the Phaser game framework. It runs entirely in a browser, and communicates with BSF using the Strophe XMPP library.

If the user allows the program access to their microphone, they can cheer or boo the actions of the agents by shouting into the microphone. Otherwise, they can simulate these actions by clicking on 'cheer' or 'boo' buttons at the bottom of the screen.

Fig. 7. A screenshot of the Punch and Judy show

show. The narrative outcomes are dependent on how the audience responds to the action, yet still conform to the rules of the Punch and Judy story world.

7 Related Work

Many approaches exist for creating interactive drama using agents as characters. Carnegie Mellon's OZ project [20] is one of the first major research efforts to use agents for interactive narrative. A dramatic structure is given to the narrative by means of a *drama manager*, which is able to see all of the actions occurring in the story world and can change anything in order to create a better experience for the user. While a drama manager rigidly enforces agents to conform to the expectations of a story, our approach differs in that it regulates the agents, allowing them some degree of agency. This allows the agents to break from the set of actions suggested by a story function. If multiple institutions present alternative sets of story functions to the agents as norms, then this means that the agents would have some freedom to change the course of the story, while the institutions ensure a coherent narrative.

Ideas from the OZ project were later developed into what became Mateas's and Stern's *Façade*, where players interact with the characters of the story through natural language. In this game, the player attends the party of a young couple (Grace and Trip) celebrating their wedding anniversary. As the course of events unfold however, the player learns that all is not as happy as it seems.

In Façade, the smallest unit of narrative action is called a *story beat*, taken from McKee's book on authorial style for screenwriters [21]. The simulation constantly monitors what the user is doing and how it may lead from the current story beat to another. Story beats have preconditions and effects on the state of the narrative, so it is the drama manager's job to work out when it makes sense to initiate a certain beat. They can be thought as analogous to Propp's story functions, but in contrast to the work presented here, operate at a lower level of abstraction, describing specific agent goals and the steps needed to achieve them. Our use of norms allows abstract story components to be written without the need to write agent goals for story events. Instead, their existing

plans are governed by institutions, and guided towards taking actions to fulfil the story components.

Riedl and Young make much use of agents and planners in their work on interactive narrative. Addressing the problem of balancing character believability with plot coherence, they present a planner-based solution for story generation [25]. Their planner is described as an 'intent-driven partial order causal link' planner, planning actions for agents in a multi-agent system based on causality and intention. In this case, *intention* refers to a component of agents based on the Belief Desire Intention (BDI) framework. Their planner observes the actions of the agents in the system and infers their intentions. While their planners have the flexibility to revise their plans based on user interactions, they still rely on an author creating all possible alternative plans that allow the agents to act out a story. Again, our use of institutions allows for story authoring on a higher level, where an author need not be concerned with writing plans.

8   Conclusions and Future Work

With our approach to interactive narrative generation, we regulate the rules of the story domain using an institutional model. This model describes what each agent is permitted and obliged to do at any point in the story. This approach alone would be too rigid, however. The audience's interactions (cheering or booing) may alter the course of the narrative, but the agents would still have blindly to follow a pre-determined set of paths. By giving our agents emotional models that change their willingness to follow the narrative, a degree of unpredictability enters each performance, giving the impression that the agents are indeed characters capable of free will. The VAD emotional model is essentially reactive and quite simplistic. A more deeply-rooted emotional response, perhaps better aligned with BDI, could be achieved, based on the appraisal theory of emotion. We also note that Propp's formalism is very specific to Russian folktales and could be replaced with something more general, such as Lehnert's story functions [16]. Cavazza et al. [6] suggest the use of Linear Logic as a promising possibility for narrative formalisation, with Martens et al. [19] exploring its use further.

In our current approach, we describe the use of institutions for the modelling of narrative. There are two major unconsidered aspects in the current implementation, namely how to manage multiple scenes and how to preserve information between scenes.

The current Propp-based method is only suitable for modelling either one scene per institution or the complete narrative as a fixed sequence of scenes as one institution. Either way, the order of scenes is not governed in any way, but is pre-determined, whereas to create a truly non-linear narrative, the sequence must be changeable as the simulation or game is running. A solution would be a mechanism to coordinate multiple institutions dynamically, but still according to some higher level narrative. [12] describe the use of *metanorms* or *higher-order norms* to enable institutions to govern other institutions, which could be applied in InstAL to create a multi-tier institution for narrative.

A second issue is how to retain information between scenes: if a set of narrative events have occurred in one scene, their consequences often need to be carried over to other scenes, which implies the transfer of fluents between institutions. [17] describe a

mechanism to bridge institutions, which could be used to control the flow of events and facts between institution instances.

In meeting both these requirements, a *hierarchy* of institutions is needed to govern the narrative world: a kind of 'scene director' institution to manage the flow between scene institutions, and a means to share information in order to achieve narrative consistency and persistence. Such a combination of multi-tiered and co-operating institutions offers the potential for the generation of a much richer way of regulating narrative events in a multi-agent system. The simple proof-of-concept set out in this paper provides a first step towards developing an institutional model for narrative as well as a preliminary validation of the WIT model.

References

1. Ahn, J., Gobron, S., Garcia, D., Silvestre, Q., Thalmann, D., and Boulic, R. An NVC emotional model for conversational virtual humans in a 3D chatting environment. In *Articulated Motion and Deformable Objects*, pages 47–57. Springer, 2012.

2. Artikis, A., Sergot, M., and Pitt, J. Specifying norm-governed computational societies. *ACM Transactions on Computational Logic (TOCL)*, 10(1):1, 2009.

3. Blanco-Vigil, P. C. N. *Agent mediated auctions: the fishmarket metaphor*. PhD thesis, Universitat Autònoma de Barcelona, 1998.

4. Bordini, R. H., Hübner, J. F., and Wooldridge, M. *Programming multi-agent systems in AgentSpeak using Jason*. John Wiley & Sons, 2007.

5. Cardoso, H. L. and Oliveira, E. Institutional reality and norms: Specifying and monitoring agent organizations. *International Journal of Cooperative Information Systems*, 16(01):67–95, 2007.

6. Cavazza, M., Champagnat, R., and Leonardi, R. The iris network of excellence: Future directions in interactive storytelling. In *Interactive Storytelling*, pages 8–13. Springer, 2009.

7. Cliffe, O., De Vos, M., and Padget, J. Specifying and reasoning about multiple institutions. In Vazquez-Salceda, J. and Noriega, P., editors, *COIN 2006*, volume 4386 of *Lecture Notes in Computer Science*, pages 63–81. Springer, 2007. ISBN: 978-3-540-74457-3. Available via `http://dx.doi.org/10.1007/978-3-540-74459-7_5`.

8. Crawford, C. *Chris Crawford on interactive storytelling*. New Riders, 2012.

9. Fornara, N., Viganò, F., and Colombetti, M. Agent communication and artificial institutions. *Autonomous Agents and Multi-Agent Systems*, 14(2):121–142, 2007.

10. Gebser, M., Kaufmann, B., Kaminski, R., Ostrowski, M., Schaub, T., and Schneider, M. Potassco: The Potsdam answer set solving collection. *AI Communications*, 24(2):107–124, 2011.

11. Jones, A. J. and Sergot, M. A formal characterisation of institutionalised power. *Logic Journal of IGPL*, 4(3):427–443, 1996.

12. King, T. C., Li, T., Vos, M. D., Dignum, V., Jonker, C. M., Padget, J., and van Riemsdijk, M. B. A framework for institutions governing institutions. In Weiss, G., Yolum, P., Bordini, R. H., and Elkind, E., editors, *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, pages 473–481. ACM, 2015.

13. Kowalski, R. and Sergot, M. A logic-based calculus of events. In *Foundations of knowledge base management*, pages 23–55. Springer, 1989.

14. Lee, J., Baines, V., and Padget, J. Decoupling cognitive agents and virtual environments. In Dignum, F., Brom, C., Hindriks, K., Beer, M., and Richards, D., editors, *Cognitive Agents*

for Virtual Environments, volume 7764 of *Lecture Notes in Computer Science*, pages 17–36. Springer Berlin Heidelberg, 2013.

15. Lee, S., Mott, B., and Lester, J. Learning director agent strategies: An inductive framework for modeling director agents. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2011 (retrieved 9th June 2015). Online proceedings: `http://aaai.org/ocs/index.php/AIIDE/AIIDE11WS/paper/view/4107`.

16. Lehnert, W. G. Plot units and narrative summarization. *Cognitive Science*, 5(4):293–331, 1981.

17. Li, T. Normative conflict detection and resolution in cooperating institutions. In *23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, pages 3231–3232, 2013. See also [18].

18. Li, T. *Normative Conflict Detection and Resolution in Cooperating Institutions*. PhD thesis, University of Bath, July 2014.

19. Martens, C., Bosser, A.-G., Ferreira, J. F., and Cavazza, M. Linear logic programming for narrative generation. In *Logic Programming and Nonmonotonic Reasoning*, pages 427–432. Springer, 2013.

20. Mateas, M. An Oz-centric review of interactive drama and believable agents. In *Artificial Intelligence Today*, pages 297–328. Springer, 1999.

21. McKee, R. *Substance, Structure, Style, and the Principles of Screenwriting*. New York: HarperCollins, 1997.

22. Noriega, P., Padget, J., Verhagen, H., and d'Inverno, M. Towards a framework for socio-cognitive technical systems. In Ghose, A., Oren, N., Telang, P., and Thangarajah, J., editors, *Coordination, Organizations, Institutions, and Norms in Agent Systems X*, volume 9372 of *Lecture Notes in Computer Science*, pages 164–181. Springer International Publishing, 2015.

23. Propp, V. Morphology of the folktale. 1928. *Trans. Svatava Pirkova-Jakobson. 2nd ed. Austin: U of Texas P*, 1968.

24. Reiter, R. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. *Artificial intelligence and mathematical theory of computation: papers in honor of John McCarthy*, 27:359–380, 1991.

25. Riedl, M. O. and Young, R. M. An intent-driven planner for multi-agent story generation. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 186–193. IEEE Computer Society, 2004.

26. Robertson, J. and Young, R. Modelling character knowledge in plan-based interactive narrative to extend accomodative mediation. Paper presented at AIIDE workshop, 2013.

27. Russell, J. A. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161, 1980.

28. Thompson, M., Padget, J., and Battle, S. Governing Narrative Events With Institutional Norms. In Finlayson, M. A., Miller, B., Lieto, A., and Ronfard, R., editors, *6th Workshop on Computational Models of Narrative (CMN 2015)*, volume 45 of *OpenAccess Series in Informatics (OASIcs)*, pages 142–151, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

29. Tomaszewski, Z. On the use of reincorporation in interactive drama. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2011 (retrieved 9th June 2015). Online proceedings: `http://aaai.org/ocs/index.php/AIIDE/AIIDE11WS/paper/view/4106`.