University of Twente
Faculty of Computer Science

In cooperation with

University of Otago
Faculty of Information Science

K. J. Winkel

Upgrading Reactive Agents with Narrative Inspired Technology

What is the story?

Submitted in partial fulfilment of the requirements for the degree of Master of Science

Thesis Committee:

Martin Purvis (University of Otago)
Anton Nijholt (University of Twente)
Dirk K.J. Heylen
Mariët Theune
Job Zwiers

Enschede, 26th of May 2004

Abstract

Organisations of today get more and more interconnected and more and more information has to be processed. In order to be able to react quickly on this flow of information, it becomes attractive to let the organisations' interests be represented by software agents. When agents running on interconnected multi-agent systems of different organisations meet, an agent may maintain multiple interactions at once with several agents when it tries to reach its objective(s). Agent technology is needed that makes them able to deal with these (e-business) environments. Complex interactions and the unpredictability of these environments led to the idea of looking at human society for solutions. Inspired by aboriginal culture, we believe that the use of narratives enables humans to cope with complex societies. Existing literature from, and outside of, the area of Narrative Intelligence was examined in order to find some basic properties of narrative; these were used as a guideline for finding important structures and processes in this literature. The literature survey resulted in the formulation of a Narrative Framework. A part of this framework was designed further, resulting in a story model and a small implementation. Future work on the design of other parts of the Narrative Framework, the story model and the implementation are pointed out. The main goal of this thesis is to make a good case for, and start a discussion on, using narrative technology in e-business and agent design in general and to show in a concrete manner how this can be done. As a simplified model of the multi-agent environment described above, the card game Pit was used. This card game represents a simulation of a commodity-trading environment. The rules of the card game Pit and the story model were formalized using Petri nets.

Contents

Abstract 2
Contents 3
1 Introduction 1
2 Agents 5
2.1 Definition Agent 5
2.2 Intelligent Agents 5
2.3 Multi-agent systems 6
2.4 Approach taken 6
3 Artificial Intelligence 8
3.1 Different Approaches 8
3.1.1 Classical AI 8
3.1.2 Alternative AI 8
3.2 Approach taken 9
4 Narrative Intelligence 11
4.1 Introduction 12
4.1.1 Research fields 13
4.2 Narrative 15
4.3 External use 16
4.3.1 Agent-Human Communication 17
4.3.2 Agent-Agent Communication 22
4.4 Internal use 26
4.4.1 Knowledge Representation 29
4.4.2 Intelligence 36
4.5 What is the story? 42
4.5.1 Index 45
4.5.2 Fabula 47
4.5.3 Making and Manipulating Stories 50
4.5.4 Connecting Fabula and Index 54
4.5.5 Syuzhet 56
4.5.6 Overall perspective 59
5 Agent Architecture 62
5.1 Opal Platform 62
5.2 Coloured Petri-nets 63
5.2.1 Definition 63
5.2.2 JFern 67
5.2.3 Petri-net controlled Agents 69
5.3 JXTA 71
6 Pit-game Agent 73
6.1 The Pit-game 73
6.1.1 Rules 73
6.1.2 Definitions 74
6.1.3 The pit game as an e-business simulation 75
6.2 Agent Design 75
6.2.1 Original Design 76
6.2.2 New Design: Petri-Nets 80
7 Narrative Framework 100
7.1 Overall Framework 100

7.2 Scope of this thesis....105
7.3 Defining Stories & Skeletons....106
7.3.1 Key-Event....108
7.3.2 Actors / Artefacts....108
7.3.3 Fabula Structure....109
7.3.4 Stories in Petri-nets....109
7.4 Using Petri-net stories....112
7.5 Implementation....115
7.5.1 A Pit-game story....116
7.5.2 An implemented story....120
7.6 Model Trade-Offs....126
7.6.1 Actors....127
7.6.2 Key-events....131
8 Conclusions....134
8.1 Narrative Theory....134
8.1.1 Orientation....134
8.1.2 Concrete Concepts....135
8.2 Pit-Game Agents....136
8.3 Narrative Framework....136
8.3.1 Overall Framework....136
8.3.2 Design and Implementation....137
8.4 End Statement....138
9 References....139

1 Introduction

The initial inspiration for this thesis came from the differences in navigation in some non-Western cultures compared to Western cultures [39: page 251]. When navigating through large areas western navigators imagine themselves looking down on the area using maps; the Western representation of the world is a ‘paper world’. Some cultures from, for example, the Pacific Islands [14] and Australia [4] have a fundamentally different conception of space, they image themselves to be part of the space. This type of space representation is by no means inferior and is not based on maps and charts.

Chatwin gives a good example of this in his book The Songlines [4]. Chatwin lived and travelled with aborigines from Central and Western Australia and documented how aboriginals navigate through a seemingly featureless desert without reference to compasses, maps or stars. Aboriginals navigate using songs based on their myths of the creation of the land of Australia. Uttal gives a good description of how this navigation is performed [39: page 252]:

“In part, the navigation is accomplished by giving even small features of the desert symbolic meaning. Each navigator possesses an individual ‘songline’, a record of the individual’s personal cosmology. Songlines connect locations in terms of myths regarding events, or dreamings, that took place during the creation of particular features. One songline might include, for example, the story of the creation of a particular rock and the path that an ancient ancestor followed during the creation.”

Apparently there are different ways of knowledge representation equally capable and perhaps more intuitive than those of Western culture. The main point here seems to be that making an abstract model of the ‘world’, a map, doesn’t necessarily function any better than a very specific fragmental piece of information, like a *songline*. We believe that narratives lie at the base of this ‘alternative’ kind of knowledge representation.

A considerable amount of research in the area of Artificial Intelligence (AI) is done to make computers more intelligent and more intuitive to humans, but an ‘intelligent’ computer still seems very far away. We believe that the concept of ‘narrative’ will provide the missing link here.

Before continuing with the subject of narrative, first we will need to explain what the reason was for the search for this alternative kind of knowledge representation.

Organisations of today get more and more interconnected and more and more information has to be processed. In order to be able to concentrate on the bigger perspective and to deal with this continuous flow of information, it becomes interesting to let computer programs represent ones interests. Software applications must be developed that can interoperate effectively in this new, distributed, heterogeneous, and sometimes unreliable environment. Multi-agent systems are seen as a potentially robust and scalable approach to meet this challenge. Each agent on a multi-agent system is given a small subtask; together the agents look after the organisations’ interests. Agents or group of agents of different organisations can come together in a competitive environment to exchange information and services.

In these multi-agent environments interactions are complex, with an arbitrary amount of agents, which can come and go at will. The agents autonomously interacting with each other in such multi-agent environments resemble that of humans interacting in human society. In analogy with the aboriginal example above, we believe that narrative usage enables humans to deal with such complexity.

Interactions with all kinds of agents which can come and go requires an amount of flexibility and the ability to establish some kind of cooperation with one or more agents in order to be able to look after the interests of the organisation. For example, if agent 1 wants to trade Euros for New-Zealand Dollars with agent 2, but agent 2 wants English Pounds for the New-Zealand Dollars, agent 1 can decide to first trade Euros for English Pounds with a third agent. Still, it has to be able to find the third agent that wants to trade English Pounds, and maybe agent 2 already disappeared from the scene when agent 1 comes back with English Pounds. Also, if it is too difficult to get New Zealand dollars, the agent could decide to first concentrate on other objectives. Maybe the final goal of this agent was to buy sheep in New Zealand, so it could already try to make contacts with a selling agent.

To be short, anything can happen: a ‘real-life’ environment can be very unpredictable. In analogy with map-making, the research field of AI traditionally tried to cope with this by letting computer programs make abstract internal models of the world. Though, in a dynamic environment as characterized above, this internal model would be out-dated most of the time and would require huge amounts of storage capacity. We believe that narrative provides a good base for an agent to find its way in these complex unpredictable environments, structuring relevant events into a coherent whole and filtering out irrelevant events. How this can exactly be realized will be researched in this thesis.

Already some researchers have tried to introduce narrative techniques to AI, this research area was called Narrative Intelligence. Some theories look very promising, but often remain vague about specifics or lead to trivial implementations [7]. In any case, it is not clear how they could lead to solve real-world problems faced with in AI. In this thesis we want to take the area of Narrative Intelligence a step further and make a better case for the use of narrative in computer science by applying it to the multi-agent / e-business problem as stated above. This problem is e-business oriented because thesis was made in the SECML Lab of the Information Science department of the university of Otago, which is part of the School of Business. The case presented here is a continuation of earlier research done in the SECML Lab [26, 27].

In order to treat an e-business example that covers the essential issues of interest but avoids extraneous matters, we look at a ‘closed-world’ card game, Pit [15]. This card-game is a simulation of a commodity-trading environment. In order to have a suitable multi-agent version of this card game, some work needed to be done.

Already before the start of this thesis there was a (turn-based) multi-agent version of this card game, in order to make the agents truly autonomous this initial version needed to be upgraded. The main activity was to model the Pit rules in Petri-nets and enable agents to use these Petri-nets to coordinate their behaviour. On top of this ‘rule-layer’, a ‘narrative-layer’ will be added. The rule layer restricts the agent’s behaviour while the narrative-layer takes care of strategic decisions.

The use of Petri-nets was chosen because its very suitable for the multi-agent environment characterized above. With one Petri-net an agent can maintain several concurrent interactions with different agents. Because an agent has to cross organisational and international boundaries, rules and 'strategies' can change easily. Because they will be specified separately, an agent can combine different rules and strategies. For example, interaction rules can change in a different organisation, but an agent may want to use the same strategy and keep the same goals. Furthermore formalization in Petri-net allows these specifications to be analysed for, for example, deadlocks and loops.

The chapters Agents, Artificial Intelligence, Narrative Intelligence and Agent Architecture deal with the theoretical background needed for this thesis. Then the thesis is continued with the chapters Pit-Game Agent, Narrative Framework that deal with design and implementation issues. Subsequently the relations between these chapters will be explained, starting of with the most innovative part of this thesis.

The core of this thesis consists of the research on how narrative can be applied to our case. Initially we didn't exactly know how to approach this problem; this is why first a broad survey was made of the field of narrative intelligence. This survey is reported of in the sections 4.3: External use and 4.4: Internal use. In this survey a few aspects of narrative, considered important by us, and often by the researchers too, kept reappearing. These aspects were used as a guide for finding concrete concepts (for example a story structure), which can be applied directly to our pit-game agents in section 4.5: 'What is the story?'.

Finally these concrete concepts lead to a Petri-net based story model (section), situated in a larger framework, in chapter 7: Narrative Framework. This is where this thesis enters new domains: as far as we know never such a concrete story model was given which can be applied directly to a particular domain, in our case that of e-business / commodity trading. To show that this story model can be used to control the behaviour of, in our case, a Pit-game playing agent, a small implementation was given of a story within the Pit-game agent's architecture.

While the sections and chapters mentioned above constitute the core of this thesis, some background was needed. To be able to have a good sense of what an agent is and where the research area of Narrative Intelligence can be placed in that of Artificial Intelligence, chapters 2: Agents and 3: Artificial Intelligence were included before continuing with chapter 4: Narrative Intelligence. This last chapter contains the survey (sections 4.3 and 4.4) and the more concrete theory (section 4.5) in the area of narrative intelligence.

Before continuing with chapter 7 the new-design of the Pit-game agents needed to be specified in chapter 6: Pit-game Agent. In order to be able to understand this chapter some techniques, Petri-nets and the multi-agent architecture amongst others, needed to be specified first in chapter 5: Agent Architecture. Chapter 6 on the design of the Pit-game agents is quite elaborate because it accounts for a substantial amount of time spent on this project. This can be justified with earlier stated arguments concerning the use of Petri-nets.

Finally our most important findings will be summarized in chapter 8: Conclusions. It will become clear here that the research presented in this thesis is only a small start. This is why the conclusion also gives a summary of all the research that needs to be done to enable the complete implementation of our narrative framework. The main goal of this thesis is to make a good case for the use of narrative based techniques in agent design and to give a good starting point to base further research and discussion on.

2 Agents

Topics:
- What is an agent?
- What is an intelligent agent?
- What is a multi-agent system?
- How can the agents to be designed for this project be characterized using the previous definitions?

As this thesis is entirely based on the concept of agents, it is needed to give a small introduction of what is understood by agents in this thesis. First of all a definition will be given of what we understand by the term 'agent'. Subsequently it will be explained what is meant by an Intelligent Agent (section 2.2) and what constitutes a Multi-agent system (section 2.3). Finally it is discussed how this translates to the (pit-game) agents used in this project in section 2.4.

2.1 Definition Agent

According to the 'The Concise Oxford Dictionary" a definition for agent could be:

"One who or that which exerts power or produces an effect"

This definition is quite general. For example, an agent can be both a human and a computer system. In the context of this thesis by agent typically a computer system is meant. There is no universally accepted definition of what an agent in this context is. Part of the problem is, is that in different research fields different aspects of an agent are important.

To make unambiguous what is meant by an agent in this thesis, a definition by Wooldridge [40: page 15] will be used:

"An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives"

Wooldridge identifies the environment as being 'non-deterministic', which means that the agent doesn't have complete control; the agent can fail.

2.2 Intelligent Agents

Most of the time by the term 'agent', an 'intelligent agent' is meant. One usually doesn't think of an agent if it concerns a thermostat. The question is then, what makes an agent intelligent? Wooldridge answers this question by listing some capabilities an intelligent agent is expected to have [40: page 23]:

- **Reactivity:** In a dynamic changing environment the agent has to perceive and respond to changes continuously.
- **Proactiveness:** An intelligent agent has to take initiative in trying to reach its goals.

- **Social ability:** An intelligent agent can often not reach its goal by itself, the agent has to negotiate and cooperate with other agents that typically don't share the same goals.

The capabilities of Reactivity and Proactiveness are often in conflict with each other. Trying to reach a goal (Proactiveness), one can generate a plan or procedure to do this. Though, the environment can change while following this procedure. This change can possibly require a modification in the procedure or even make the goal obsolete; dealing with this requires reactive capabilities.

According to Wooldridge [40: page 24] it is not difficult to design a purely reactive agent – continually responding to its environment – neither to design a purely proactive agent – blindly executing pre-programmed procedures for reaching a certain goal. The challenge here is to design an agent that finds a balance between both capabilities. Even for humans this is a difficult task according to Wooldridge.

2.3 Multi-agent systems

In current everyday computing world single agent systems are rare. Even when a system is not interacting with other agents, it usually consists of sub-systems that must interact with each other.

Jennings defines some characteristics of a multi-agent system (MAS) [17]. These characteristics can be derived from the previously given definition of 'agent'.

- To that an agent doesn't have complete control over its environment (non-determinacy) Jennings adds that an agent also doesn't have complete information of its environment.

Very much in relation with that an agent acts autonomously, a multi-agent also has the following characteristics according to Jennings:

- No global system control.
- Data is decentralized.
- Computation is asynchronous.

2.4 Approach taken

As the title of this thesis already suggests, the original design of the pit-game agents (see section 6.2.1) was not satisfying. First of all the pit-game agents were purely *reactive* – they based their decisions only on the present.

With respect to *Proactiveness*, the agents were able to take the initiative but didn't have any structured way to reach their goals (winning the game for example). Decisions were made on the spot without any knowledge of past or future; the decisions to make in a certain (present) situation were pre-programmed (sometimes with a random factor in it).

In order to make the agents' successive decisions more coherent over time and leading to one or more goals, the agent will be needed to make decisions based on the past. Additionally the agent should be able to learn, some reoccurring events don't

necessarily have to lead to the same conclusions every time. Because the agent previously was able to see the outcome following this reoccurring event, it could decide for different action next time.

Another drawback of the original design was that it didn't have any *Social ability*. It didn't negotiate or cooperate; it only tried to reach its own goals without considering the goals of other agents. For example, it sometimes occurred that the game entered a deadlock because all the pit-game agents wanted to corner the same commodity.

This last drawback can be viewed in different levels. On the lowest level an agent has to try to synchronize its communication in some way with the other agent. In the case of the pit-game (see chapter 6) waving 10 different cards in one second won't allow any other agent to react. On a higher level there is the example of the deadlock problem. On an even higher level an agent can, for example, make agreements with other agents not the trade with a certain agent.

Within the time constrains of this thesis it was not possible to make a new design of the pit-game agent which incorporates all these new features. Instead, these requirements were used as a guideline to what kind of agent the work in this thesis should finally lead to. The proceedings of this thesis should be at least promising as a means to fulfil these requirements.

The first priority was to make the agents' behaviour more coherent over time. Its social abilities were of second importance, though, as will be seen in the next chapter, the area of NI is very much related and inspired on human social behaviour.

At last, the original design of the pit-game agents didn't constitute a true multi-agent system. There was a global system control, and communication was synchronous; the agents weren't truly autonomous. This had to be changed too and is further discussed in chapter 6.

Summary:

- Definitions where given of 'agent', 'intelligent agent' and 'multi-agent systems'.
- Primary design guidelines: make agent behaviour more coherent over time, make agents truly autonomous.
- Secondary design guidelines: agents need to be able to learn from past experiences, agents need social abilities.
- Narrative Intelligence is inspired on human behaviour.

3 Artificial Intelligence

Topics:

- The two main movements within Artificial Intelligence will be identified: Classical AI and Alternative AI.
- The research area of Narrative Intelligence will be situated within both movements.

Already a lot of agent architectures have been designed to deal with more or less of the aspects discussed above. A small overview of two different approaches within Artificial Intelligence will be given in section 3.1. Finally Narrative Intelligence and the approach used in this thesis will be situated within these two approaches in section 3.2.

With regard to the previous chapter it has to be noted that though most agent architectures draw upon some AI techniques of some sort, most of it is standard computer science.

3.1 Different Approaches

There are two major trends of thinking within current research on Artificial Intelligence. Both trends can be given a lot of different names, here it is chosen to name them Classical and Alternative AI. A short characterization of both trends will be given.

3.1.1 Classical AI

Classical AI is the oldest school within Artificial Intelligence and the majority of researchers still follow this school although they are trying to expand their focus.

In classical AI an agent is constructed and functions in a top-down manner. The main focus is on maintaining an internal (symbolic/abstract) representation of the world, decision making is based on manipulating these presentations. The agent is symbolically grounded.

Referring to the capability of *Proactiveness*, a Classical AI agent would strictly follow its procedures and plans to reach its goals. All ‘unexpected’ situations should be accounted for in these procedures and plans, otherwise the agent wouldn’t know what to do.

According to Sengers [37] this way of thinking was mainly influenced by (American) culture. The mind was viewed as being separate from the body, also called the ‘schizophrenic’ model of consciousness. In the last 3 decades people started viewing mind and body as being inescapably interlinked. According to Sengers Alternative AI is clearly inspired on this new way of thinking.

3.1.2 Alternative AI

The smaller, newer school of Alternative AI considers the views of Classical AI as fundamentally wrong.

In Alternative AI an agent is grounded in its body (embodied) and environment. Intelligent behaviour is considered a product of the interaction of the agent with its environment; this is often called *Situated* Intelligence.

Another complementary view within Alternative AI is that intelligent behaviour emerges from interaction of various simpler behaviours; an agent simply reacts to its environment without reasoning about it. This corresponds with the former mentioned capability of *Reactivity*. One of the research fields within Alternative AI that incorporates these views is Artificial Life.

In Alternative AI an agent is typically built bottom-up. For example, a certain set of situations is identified and for each situation certain behaviour is being determined, these 'behaviours' can be given priorities. When seen in action this agent can appear as quite intelligent though it's only following a few simple rules. This agent can be called purely reactive.

3.2 Approach taken

The focus in this thesis lies on Narrative Intelligence, a relatively young sub field of Artificial Intelligence.

Dealing with a multi-agent system with the properties described in section 2.3 it seems logical to look at human societies for inspiration. A complex artificial society where lots of agents interact autonomously is very similar to human society. In this thesis the role of narrative in human societies will be given the focus of inspiration, for reasons already explained in the introduction (chapter 1).

Narrative Intelligence can be approached from both a Classical and Alternative AI perspective. For example, Sengers and Dautenhahn have a more Alternative perspective on Narrative Intelligence while Schank[1] is more influenced by Classical AI.

In a way Narrative Intelligence can be viewed as being more Alternative AI oriented. As can be seen in the example of the aboriginal stories (in chapter 1) these narratives comprise a lot less information than a whole map, they are very fragmental and meaningless when not used in the Australian desert. The narrative is grounded in its environment.

Still, narrative provides some representation of the environment and this way Narrative Intelligence can be seen as a good compromise between the schools of Classical and Alternative AI.

In this thesis the stance is taken that neither pure Classical nor pure Alternative AI is the solution, though the views of Alternative AI are favoured more over those of Classical AI. Narrative Intelligence seems to fit very well with this stance.

[1] Note that the authors Sengers, Dautenhahn and Schank will be extensively discussed in the following chapter.

Summary:

- Narrative Intelligence provides a good compromise between the movements of Classical and Alternative AI.

4 Narrative Intelligence

In this chapter the new field of Narrative Intelligence will be given a closer look. In the Introduction it will be discussed how the field of Narrative Intelligence came to life. This is followed up by an overview of research fields and researched involved in the area of Narrative Intelligence, particularly those used in this project.

In order to have an idea about what can be understood by narrative and to give a foundation for how the rest of chapter 4 will be approached, section 4.2: Narrative was included next. The rest of the chapter is divided in the sections 4.3: External use, 4.4: Internal use and 4.5: 'What is the story?'.

The sections 4.3 and 4.4 are important because in these chapters it will be explored what the essential properties of narrative are. In both chapters this was often done by trying to find the origins of narrative, how narrative did evolve in humans.

The chapters were split up in external and internal, because some researchers mainly focussed on narrative in communication and behaviour, like Sengers and Dautenhahn. Others more on narrative for internal purposes like knowledge representation and intelligence, of which Schank is a good example. The cause of this is mainly that Sengers and Dautenhahn work in the field of Alternative AI, where appearance and outer behaviour gets more attention. Schank is influenced by the school of Classical AI where internal processes get more attention.

Section 4.3 was subsequently divided into Agent-Human (section 4.3.1) and Agent-Agent communication (section 4.3.2). The main inspiration for this was that for example Sengers looks at agents from a Human-Computer Interaction point of view; an agent only has to communicate with humans. Because in this project artificial agents have to communicate with each other the idea rose that this communication could be different. Because Sengers work nevertheless is very valuable for this project it was included in section 4.3.1. Because of differences between human society and possible artificial societies, like for example language, more basic properties were looked for in section 4.3.2. The latter was done on the basis of Dautenhahn's work.

In section 4.4 it was mainly tried to find good motivations for the idea that narrative is not only something that structures for example communication but also has a profound impact on internal representation (section 4.4.1) and intelligence (section 4.4.2), both internal processes. Using literature from Schank and Dennet a good and complete as possible characterization was given of how narrative might have its purposes internally.

In the above two chapters, Internal and External Use, and its subchapters, regularly some main points of interest are discussed. Examples of these points of interest are: Communication, Coherence, Language, Culture, etc. These were treated specifically because they were considered particularly important subjects and often turned up as being so across different literature.

Culture is for example important because of our initial inspiration of aboriginal songlines which are actually stories belonging to aboriginal culture. Every next chapter often sheds more light on the points of interest of the previous chapter.

Finally these points of interest were used as a basis for the rest of this thesis and especially for finding more concrete ideas about what a story is in section 4.5. The idea here was to find theory that could directly be used in agent design and possibly implementation. For this the work of Schank and Bordwell proved especially useful.

Points of interest like ‘Interpretation’ and ‘Behind the story’ were included because of a chapter that was later omitted from this thesis because of time restrictions. This chapter was supposed to follow-up ‘What is the story’ and would look for influences on the interpretation of a narrative other than the presented narrative itself and if these influences would be the same for humans and artificial agents.

Although chapter ‘What is the story’ gives reasons to believe that a narrative itself (the syuzhet) suggests quite an unambiguous interpretation, it still seems that there are some loose ends. In the chapters Intelligence and ‘What is the story?’ the importance of having interests comes up several times and in chapter ‘What is the story’ it turned out that especially with the selection of so called ‘skeletons’ the theory of Bordwell and Schank was inadequate. In the conclusion some recommendations are made for further research on this problem.

For this reasons, and to give a more open and complete view on narratives these points of interest and comments on this subject will be kept in the following discussion of the theory.

4.1 Introduction

Topics:

- The origin of the research field of Narrative Intelligence.
- Research fields related with Narrative Intelligence.

The term Narrative Intelligence came to life when two graduate students at MIT Media Lab started the Narrative Intelligence Reading Group [9] in the fall of 1990. One (Marc Davis) was a humanist (literature, philosophy and language) who wanted to build programs that could automatically assemble short movies from archives of video data. The other (Michael Travers) was a computer scientist wanting to program software agents that could understand a simulated world, each other, and themselves.

The problem was that both of their disciplines seemed to be too restricted to situate their work in. Both the areas of Classical and Alternative AI within Artificial Intelligence seemed to lack a coherent model of representation, while literary theory seemed uninfluenced by the theoretical roots and progress in computational technology.

While the areas seemed to have common issues it was not that straightforward to find common ground between these areas. In both areas different terminology was used to talk about things, core ideas like “representation”, “language” and “communication” meant different things to them. Also standards and practices for what constituted acceptable talking, reading, writing, analysis, presentation and production (text and artefacts) were all quite different.

The Narrative Intelligence Reading Group started off as a student-run reading group with no curricular or departmental guidelines to adhere to. The reading group grew with members from different other universities, extending the number of research fields. After several years an interdisciplinary methodology for Narrative Intelligence emerged and the reading group started to have its impact on MIT's curriculum. The Narrative Intelligence Reading Group existed for six years, after that the group kept functioning as a mailing list.

The founders of the group identify that there is still much work to be done, the research field of humanities still looks at computation as a mere instrumentality and most computer science programs do not offer courses in which literary and media theory are taught and applied.

4.1.1 Research fields

Core works used by the Narrative Intelligence Reading Group originate from the following fields. The first two disciplines were part of the group from the start; later the group broadened its view to the others:

- Artificial Intelligence & Cognitive Science
- Literary Theory
- Media Studies
- Psychology & Sociology
- User Interface Theory
- Software
- Social Computing
- Constructionism in Science and Learning

Using somewhat the same division of disciplines, areas that have been studied for this thesis to get more insight in concept of Narrative Intelligence are:

- Computer Science / Information Science
    - Artificial Intelligence
    - User Interfaces
    - Story Generation
- Cognitive Sciences
- Literary Theory
- Media Studies
    - Screenplay / Scriptwriting
- Mathematics
- Philosophy
- Psychology
- Culture

Artificial Intelligence, User Interfaces and Story Generation are all grouped under Computer Science as literature used from these disciplines have a Computer Science orientation. Computer Science was the first area to be researched as the main goal of this thesis is to come up with some concrete ideas for implementation of narrative technology.

Under the rather broad discipline of Artificial Intelligence most of the work researched was produced by Dautenhahn and/or Nehaniv *(Ref?)* as their work seemed most applicable to our work and they seemed to have a modern approach to AI (Alternative AI). Also some more old-fashioned approaches (Classical AI) to AI were examined briefly (BDI, Planning).

Closely related with AI, the areas of User Interfaces and Story Generation seemed inspiring fields of research. In the area of User Interfaces the work of Sengers (and Mateas) seemed most related to our work, as she tries to apply Narrative Intelligence to user interfaces and seems to have put significant effort in this field [20, 36, 37, 38]. For well-defined algorithms of story creation the area of Story Generation was examined in a somewhat arbitrary way, as a lot of ideas exist on how to generate stories by computer.

In addition to Computer Science the view was broadened to other related sciences, keeping in mind that the most important was to find inspiration for an implementation. As it was not possible in the given time-frame to first read all important basic works (for example those specified in [9]) related to Narrative Intelligence and then start finding hints on possible implementations this work is mainly concentrated on the latter.

The area of Cognitive Sciences covers a lot of disciplines similar to those of Narrative Intelligence. One of the main influences on this thesis, and in the scientific world, is the work of Schank. Schank's work is situated in the fields of AI, Education and Psychology. He became known by his work on Scripts Plans and Goals [31] and Case-Based Reasoning (CBR) [32]. CBR is closely related to Narrative Intelligence and could be useful, but especially his more recent work [33, 34] relates very well to the subject of this thesis.

Literary theory and Media studies are closely related: both are studying recorded narratives. In literary theory the most recent work we found was by Abbot [1], which seemed to be too much of an in-depth structuralist work to be of direct use. Media Studies seemed to have a wider perspective on narrative. Especially the work of Bordwell [2] got preference because they seemed to focus more on interpretation (cognitivism) then on structure.

Mathematics maybe don't seem to be related to Narrative Intelligence, but can provide models of memory (Nehaniv [21]) and possibly other concrete definitions (logic, algebra) applicable to Narrative Intelligence. This field of science is mentioned primarily because of Nehaniv's work, which gives some interesting definitions of autobiographic memory. In this project Petri-nets (section 5.2) were chosen to model narratives, but Nehaniv's work provides an interesting alternative.

To get a wider perspective on narratives some additional works were examined in the fields of Philosophy, Psychology and Culture, the latter because our initial inspiration came from aboriginal culture (see Introduction). In the next chapters, several some somewhat arbitrarily chosen but applicable views from Psychology (Schacter, Bruner) and Philosophy (Dennet, Carr) on narrative are touched upon to give Narrative Intelligence and this thesis some perspective.

Summary

- The research field of Narrative Intelligence originates from the Narrative Intelligence Reading Group started by Marc Davis and Michael Travers at MIT Media Lab.
- Narrative Intelligence tries to find common ground between humanities and computer science and covers a large number of research fields.

4.2 Narrative

Topics

Some initial perspective on narrative by answering the following Psychological/Philosophical questions:

- Where is narrative?
- Is narrative important for humans?
- Is there a difference between told and experienced stories?
- Does there need to be a teller and audience?

Before continuing going deeper into existing theories and works on Narrative Intelligence, it is important to ask ourselves: "where we actually find narrative?" and "where does narrative start and reality end?"

The first things that come to mind are: telling, reading and writing a story. Possibly watching a movie or even making a movie. But often also paintings are interpreted as narratives as Abbott [1] gives some good examples of.

These narratives are already pre-made by humans, but what about the world around us? Up to which extent is the daily life interpreted as narrative? That humans have a natural tendency to interpret is argued by Schacter [30]. However, as most psychologists, he doesn't attribute narrative having a constituting role in human memory. Someone who does that is Schank, he argues "that stories about one's experiences and the experiences of others are the fundamental constituents of human memory…" [34: page 1].

It is also not that obvious to make a distinction between dealing with pre-made narratives and constructing 'first-order' narratives directly from the world around us. You could argue that the difference between both is that pre-made narrative has a sender, a maker having a possible message and that the world around us doesn't have any meaning in particular. Constructivists like Bordwell argue that this isn't the case, which suggests that analysis of written stories and movies can be extended to situations where narratives are not particularly made. This would be the case for a pit-game agent, only perceiving events in and around the agent itself (see chapter 6).

Another promising argument for this is that of philosopher David Carr. He observes that "a strong coalition of philosophers, literary theorists, and historians has risen up of late, declaring … real events simply do not hang together in a narrative way…" [3: page 7]. Carr is less sceptical about narratives, "…its structure inheres in the events themselves. Far from being a formal distortion of the events it relates, a narrative account is an extension of one of their primary features" [3: page 8]. As a first

argument for this he states that what is considered ‘real’ doesn’t have to be the physical world, a mere sequence of random events. For this he cites Husserl *(11)*, who says that even the most passive experience involves retentions of the past and protention (anticipation) on the future, which you could call ‘human reality’. This moves ‘reality’ closer to a structured narrative. He continues his well-constructed argument with saying that also the beginning-middle-end structure is not an uncommon thing in real life, giving birth and death as an example.

In contrary to Bordwell he thinks a storyteller and audience belongs to the concept of story. An individual may tell stories to himself, sometimes assuming the point of view of audience, adjusting the story to the events or adjusting the events to the story *(16)*.

This doesn’t represent a complete survey of views on narrative by and large, but it gives some perspective on narratives in general. It also gives us some clues on how to approach the rest of our research on Narrative Intelligence.

First the area of Narrative Intelligence is broken up in research that is done on communicative utilisation of narrative (section 4.3: External use) and research that is done on more individual goals of narrative like knowledge representation and comprehension (section 4.4: Internal use).

After this it is tried to find some important aspects of narrative possibly usable for implementation in section 4.5: ‘What is the story?’.

Summary

- Narrative can be found everywhere around us.
- Narrative possibly plays a constituting role in human memory.
- Possibly no difference needs to be made between told and experienced (‘first order’) narratives.
- It (yet) remains undecided whether or not a storyteller and audience are needed.

4.3 External use

As said in chapter 2 a human can also be called an agent, it was decided that with the term ‘agent’ a computer system is meant. This distinction is useful here because narrative communication involving human agents is possibly a more restricting view then narrative communication between any type of agent, which will be discussed in section 4.3.2.

4.3.1 Agent-Human Communication

Topics

- Narrative purely as a communicational tool which only has a function in the appearance of an agent, investigated on the basis of Sengers research on Human Computer Interaction.

A lot of work has been done on making artificial agents communicate with humans to make interaction with them easier. Several researchers in the field of human-computer interfaces argue that narrative should be used as a basis. If humans often make sense of the world by assimilating it to narrative, they argue, it makes sense to design our systems in a way that allows people to use their narrative skills in interpreting these systems [20: page 3].

Work in this field doesn't apply in a direct way to this thesis. In one way because it involves communications with humans in specific, which introduces some restrictions as can be seen in section 4.3.2.

Citing Bruner, Sengers suggests that people understand and interpret intentional behaviour by organizing it into a kind of story [38: page 3]. She identifies *Narrative Diachronicity* as the most basic property of narrative, which means that narrative relates events over time. Currently behaviour-based agents re-decide the best action the agent can take continuously; this way behaviour-based agents seem to display a kind of "schizophrenia".

An user-interface agent using narratives only has to *appear* as having some coherence, which Sengers does by letting the agent generate 'narrative cues' to support users to generate a narrative explanation. Human-computer interface design doesn't impose restrictions on the agent's internal design, the agent doesn't have to understand or model even in the most simplistic way what is happening. A human-computer interface already succeeds if it makes interaction with a human somehow a bit easier.

An autonomous agent possibly even representing people's interests should be able to derive more far-reaching abilities from narrative, like behaving 'intelligent' and being flexible. It stays arguable however whether or not narrative actually directs internal representation and comprehension in addition to communication, which will be explored further in section 4.4.

Note that coherence is identified as intentional coherence by Sengers. This implies intention is the main aspect which makes the story important (for a human), which remains to be seen (see next chapters).

Despite of this, Sengers gives a very useful interpretation of Bruner's narrative properties with respect to agent design [38] mostly focussed on the agents appearance, the first property was already mentioned.

- *Narrative Diachronicity*

Narrative relates events over time. Currently behaviour-based agents re-decide the best action to be taken continuously.

- *Particularity*

  Sengers (citing Bruner) gives the rather strong argument that narratives are always about particular events and individuals. Sengers still values modelling at an abstract level in order for an agent to be able to behave autonomously, but this seems to be more an aspect of an agent's internal design. Particularity seems to be assigned more to communication, the agent's appearance. She gives some good examples that particularity in appearance is indeed important for an agents' believability, at least to humans.

- *Intentional State Entailment*

  When people are acting in narrative, the important part is not what the people do, but how they think and feel about what they do. Sengers argues that agents should at least appear to be thinking about what they are doing: agents should express reasons for their behaviours. Currently agents don't have access to their reasons because they are part of the implicit architecture of the agent.

- *Hermeneutic Composability*

  Narrative is understood as a type of communication between author and audience. Events of a narrative are not understood individually but in the context of the other events and the story as a whole, which is a complex and circular process. Sengers says that agents which parts are designed completely separate are bound to end up misleading the user. This stands in contrast to the currently fashionable Alternative AI approach where interrelationships may emerge from separately designed pieces – Sengers says it may be the best we can do.

- *Canonicity and Breach*

  Seemingly in contrary to the *Intentional State Entailment* argument, Sengers states that narrative should not always be easy interpretable and predictable, it should contain something unexpected. Users are very good at creating narrative; stereotyped actions bore the audience.

- *Genericness*

  Narratives are understood with respect to genre expectations, which we pick up from our culture. Current practice of building agents should consider the context in which the agent will be used. It is even argued that cultural baggage of researchers already affects the way agents are designed. AI researchers should be aware of the relationship of their research and their culture and society as a whole. A good example is the origin of two main traditions of AI, classical and alternative AI (see section 3.1).

- *Referentiality*

An agent doesn't have to have an objective model of the world to keep track of what is going on. It only has to keep track of its current viewpoint and goals. A plausible narrative does not essentially refer to the facts in the real world, but has to stand up to its own subjective tests of realism. Sengers makes the interesting argument that what the agent really *is* doesn't have to be more than the impression it makes. This in contrary to the common viewpoint in AI that an agents 'real' essence is identified with an agents' internal code, often resulting in 'incomprehensible' agents. Note that this is with respect to humans and the notion of making humans believe the agent is alive.

- *Normativeness*

  Narratives are strongly based on the conventions that the audience brings to the story. While breaking conventions as argued in *Canonicity and Breach*, they still depend on those same conventions to be understood and valued by the audience.

- *Context Sensitivity and Negotiability*

  Agents should not be built to provide pre-packaged narratives to the user. How the user interprets the narrative cannot be enforced but only negotiated, the agent should only provide narrative *cues*. The user interprets narrative with respect to his own lived experience. As Sengers says eloquently: "Narrative is the interface between communication and life".

- *Narrative Accrual*

  Sengers argues narratives are linked over time, even saying that in this way they form culture or tradition. They don't have to adhere to one principle or larger paradigm. Stories can be in contradiction with each other. Sengers acknowledges that these mechanisms are not well understood.

She continues by arguing Artificial Intelligence inherited scientific research traditions which properties are exactly the opposite of properties of narrative.

The narrative properties are interpreted from a human-computer interface point of view. Narrative properties are seen as something that can make the agent look more alive to a human. For this, agents should 'appear' particular, express reasons and behave unexpected (to a certain extent).

In addition to this, Sengers interpretation of narrative properties gives rise to some important issues:

Abstraction versus Particularity

The property of particularity is especially needed in the case of interpretation by humans. Sengers acknowledges that abstraction is still needed in the agents design, as an agent cannot know in every detail what will happen. Particularity seems to be more a property of communication.

Internal versus External

The former leads us to the question: to which extent are narrative capabilities only appearance and to which extent do they actually affect the agent's internal design? Or more general: what should be modelled internally and what not.

In *Referentiality* Sengers is trying to shift from the more conservative AI researcher's view that the 'real' essence of an agent is in its internal code to giving more importance to an agent's impression. Although this position is coloured by a human-computer interface design perspective, this is an important issue.

The latter is in correspondence with the current critiques on Classical AI of maintaining an objective world model (see chapter 3). An argument that favours some central internal representation is that made in *Hermeneutic Composability*. She says that current modular design of agents is in contrast with stories that have to be understood as a whole. However, she doesn't discard the Alternative AI approach of emerging interrelationships from separately designed pieces.

This issue was the cause to separately look closer at narrative as an external, communicational means in this section and narrative for internal use in section 4.4.

Interpretation

A lot of the narrative properties deal with interpretation, as Sengers work is in human-computer interfaces this would be human interpretation. In *Canonicity and Breach* she talks about not unexpectedness of events.

Later it becomes evident that expectedness is not something pre-defined. In *Genericness* she argues that culture raises expectations and in *Normativeness* it is said that what is expected and conventional depends on the audience.

In *Context Sensitivity and Negotiability* she says interpretation of a narrative cannot be enforced but is negotiated in a complex way. This implies that there is a sender, who wants to 'negotiate' a message to the audience. The latter is in contrast with the constructivist approach, like Bordwell's (see section 4.5). It remains to be seen how much control there is over interpretation of a story, if there is any.

Also Sengers work assumes a human interpreter, which is not present in the case of this thesis. This is the reason that we will investigate some more general applicable theories in the next chapter.

Culture

Last but not least the concept of culture is mentioned. The fact that culture raises expectations is already mentioned. In *Narrative Accrual* she gives an idea of how culture could arise from stories by linking smaller stories into a bigger story that could become part of culture or tradition. This will be also further discussed in the next chapter.

Summary

- Narrative cues the interpretation of a human: narrative provides coherence between the actions taken by an agent; this makes the agent easier to understand by a human.
- Sengers discusses important properties of narrative; these properties originate from Bruner's work and are often referenced (used) within the field of Narrative Intelligence.
- The fact that Sengers puts emphasis on appearance leads to the question whether or not narrative only structures communication or also structures internal processes in the human mind. This led us to dividing our initial survey (see Introduction) into sections External Use and Internal Use.
- The fact that Sengers work is very much oriented on human interpretation leads to the idea that this might be too specific when Sengers' work needs to be applied to an (artificial) agent society. In turn, this led us to dividing section External Use into sections Agent-Human and Agent-Agent communication.
- It is believed that narrative plays an important role in culture.

4.3.2 Agent-Agent Communication

Topics

- Going beyond communication with humans more basic properties are looked for in the use of narratives by primates, this is done using Dautenhahn's work on her "Narrative Intelligence Hypothesis".

In this chapter the scope of narrative communication is extended to theories that have a wider view than that of human-computer interaction. The roots of narrative will be searched for in the context of communication. Humans are still the main inspiration for this, but other 'social animals' can provide us with more basic properties of narrative that could give a better view on its essence. This will be discussed with Dautenhahn's research as a guideline, a researcher who has put a lot of effort in this field.

Dautenhahn's view on Artificial Intelligence is influenced by her (Biological) Cybernetics background. She emphasises her work on embodiment, social interaction and autobiography in the design of (robotic) agents. In 1999 she published her first work [6] in the area of Narrative Intelligence [16], which is related with her work on autobiographic agents.

This work is a first proposal of her Narrative Intelligence Hypothesis (NIH) [8], which Dautenhahn bases on the Social Intelligence Hypothesis (SIH), also called Machiavellian Intelligence Hypothesis or Social Brain Hypothesis. The SIH tries to find origins of human intelligence in terms of the evolution of primate social interaction and intelligence.

Many mammal species live in highly individualized societies. In individualized societies group members recognize each other individually and interact on the basis of historical interactions. Preserving social coherence, and cooperating and competing with group members produces a complex social field. The SIH suggests that primate intelligence evolved in order to cope with this social complexity, which resulted in an increase of brain size. This increase of brain size in return resulted in an increased capacity to further develop social complexity. Even later in evolution human intelligence extended to solve problems outside the social domain according to the SIH.

The question is, why did human 'apes' in particular evolve with more sophisticated mental skills? The NIH says that communicating in stories co-evolved with increasing social dynamics (see SIH) because narrative is particularly suited to communicate about the social world. It proposes that narrative is used in particular to communicate about third-party relationships (for example gossiping). In this, the evolution of language played an important supporting role as a way of communicating narrative, while non-human primates kept using social grooming.

Summarizing, in order to make artificial agents storytellers they need language and social intelligence. According to Dautenhahn [6: page 6] a story-telling agent should have the ability to:

- *recognize* individuals,
- *understand* others (empathy / social skills),
- *predict* behaviour of others and outcomes of interaction (need experience),
- remember and learn interactions with others and to build *direct relationships,* and
- remember and learn interaction between others, to understand *third-party relationships* (gossiping).

It is evident that social skills and communicating about social life play a major role in Dautenhahn's view of story telling agents. In one of her more recent papers [8] she expands the NIH by specifying stages in which story-telling evolved:

1. Non-verbal, physical social grooming (non-human primates).
2. Non-verbal, enacting stories in transactional narrative format.
3. Using language and verbal narratives.

In the first stage only one-to-one communication was possible. In the later stages one-to-many communication was enabled, first through non-verbal 'enacted' stories which allowed for a higher social complexity. The use of language reduced the amount of time needed for communication, allowed for communication about third-party relationships and even bigger social groups. Also, language features documentation, transmission of knowledge to next generation and communication between geologically separate locations.

Narrative got an important role in the use of language, because it gave language a format which is particularly suited for social communication, Dautenhahn shows that narrative structure is much related with the format of physical grooming but is more flexible. In addition to grooming, narrative can include (fictional, historical) characters which are not present at the moment of telling. Also narrative can convey sensual, emotional and meaningful aspects.

Subsequently, Dautenhahn tries to give suggestions for a possible preverbal transactional format of narrative. This preverbal format could provide important insights into a possible narrative format. Although Dautenhahn is still searching for a more elaborate form, she suggests a transactional format [8: page 256] identified by Bruner & Veltman as a starting point. This research, done in the context of narratives and autism, identifies four stages in a preverbal transaction:

- canonical steady state
- precipitating event
- a restoration
- a coda marking the end

Dautenhahn shows that this transactional format is of wider importance by identifying it in the social behaviour of chimpanzees. In order to elaborate this transactional format she suggests looking at various social organizations in primates and pre-primates. The transactional format can be influenced by group size or by other aspects of specific animal societies (types of interaction, roles of group members).

In section 4.5 it is further discussed what actually is the most important in a narrative in terms of structure and other aspects already touched upon here. Some issues relating to Dautenhahn's work need to be given a closer look, putting them in perspective of the previous and next chapters:

Social Communication
It is apparent that Dautenhahn doesn't only view narrative from a communicational perspective, but particularly situated in a social environment. Dautenhahn searches for a transactional format in social interactions and suggests that narrative is particularly suited to communicate and deal with the complexity of social life. This is why she tries to find the roots of narrative in social interaction, resulting in a proposal for an initial preverbal transaction format.

By specifying this transactional format she seems to distillate some important aspects of narrative. On the other hand, it can be argued that she still views narrative too much in a human oriented way. An earlier work on the concept Social Intelligence [5: page 6], which she closely relates to Narrative Intelligence, shows that she focuses specifically on 'human-style' social intelligence:

"Human-style social intelligence can be defined as an agent's capability to develop and manage relationships between individualised, autobiographic agents which, by means of communication, build up shared social interaction structures…"; "…I use the term social intelligence always in the context of human-style social interaction and behaviour."

Interpretation
Although humans and primates are our only possible inspiration, it could prove that for artificial intelligent agents the social aspect of Dautenhahn's NIH doesn't apply. An artificial agent 'living' in an artificial world could have different drives than social or other human drives (emotions). An artificial agent can be 'grounded' in another type of world in which 'social' as we know it doesn't exist or exists in a different way.

In favour of narrative intelligence based on human-style social intelligence is that it could not be desirable to let agents evolve or interact in a totally different way from humans. A major reason for the design of intelligent agents is to let agents act in behalf of humans, without the same (social) grounding they could not be able to do this.

While research on the project progressed, one of the main properties of narrative turned out to be finding coherence between smaller and larger (culture) groups. Furthermore, one of the initial reasons to look at human society was humans' ability to deal with interactions in complex, large, societies (section 3.2). So, human-style social abilities might be exactly what are needed for artificial agents in order to get them to be able to cooperate in large groups.

Individual Usage
Though Dautenhahn looks at narrative as basically being a (social) communicational tool, she acknowledges that narrative is also used for individual purposes [8: page 261]. Telling stories to oneself is important for making meaning of events [8: page

254, 255]. She says that *at least* for humans stories are most effective in a communicational and social context.

Language
Dautenhahn's proposal to a preverbal transaction format provides an abstraction from specific human language; this in contrary to most works in Literary Theory. This abstraction could prove useful in an artificial agent environment where other or no language can be used, as is the case in the context of this thesis. As already said, in Dautenhahn's evolutional argument language plays a very important role, language makes communication very efficient and can be used for one-to-many communication.

Culture and Imitation
The transactional format of narrative is based on 'culturally canonical forms of (human) action and interaction" [8: page 256]. Imitation seems to play an important role in passing on these canonical formats to children. The most interesting aspect here is that culture, in this case human culture, seems to have an influence on the (preverbal transactional) narrative format. Culture here seems to form narrative in addition to Sengers'/Bruner's interpretation of culture, where narratives seemed to constitute culture (section 4.3.1 / Narrative Accrual)

In this chapter narrative is being viewed upon as stemming from (social) communication. While Dautenhahn acknowledged narrative also is used for individual purposes, it could even be that narrative finds its origin in the individual. In the next chapter this perspective on narrative will be given a closer look.

Note that in terms of possible applications of narrative theory to our specific case – the pit-game – it could be beneficial if communication proves not to be the major aspect in narrative. There are no facilities providing a way of explicitly communicating narratives in the original design of the pit-game agents (see section 6.2.1).

Summary

- **Social Intelligence Hypothesis**: intelligence evolved in order to cope with increasing complex societies.
- Social relations were first maintained through physical grooming; enactment, language and narrative provided more advanced ways of communicating.
- Already in the phase of enacting stories, there was a certain standard way of performing social transactions: the preverbal transaction format.
- **Narrative Intelligence Hypothesis**: language provided a means to communicate faster and to more individuals at once. The preverbal transaction format structured the use of language, which resulted into narratives. This enabled more complex societies and in turn resulted in the need for more intelligence.
- It could be possible that other transaction formats are more suitable in an artificial world. Though, the preverbal transaction format enabled humans to cope with complex societies, which suggests that it could benefit artificial societies in the same way.
- The preverbal transaction format abstracts from language, which gives better insight into the essence of a narrative and could help with the translation of human narrative to artificial worlds.
- Culture influences the preverbal transaction format.

4.4 Internal use

Topics

- In the previous chapter narrative is basically seen as something with essentially external purposes. In this introduction a good argument is made that narrative not only structures social behaviour.

In the previous chapter Dautenhahn gave social communication as the evolutional origin of narrative. In contrary to this Dennet gives an evolutional origin with more internal purposes, this will be discussed before continuing with Schank's more practical theories on knowledge representation and intelligence. In this introduction it will also turn out that Dautenhahn's theory and Dennett's are not necessarily incompatible.

Dennet [10] argues that every human seems to be a "virtuoso novelist". According to him the human brain consists of a bundle of autonomous parts that can cause our behaviour to be disunified at times. Humans always try to find coherence by making it into a 'good' story, the autobiography. As Dennet says "we put the best faces on as we can". The main 'fictional' character in this autobiography he calls the 'self'. Dennet does not particularly mention whether or not this self also constitutes self-consciousness.

In this he finds an analogy with the concept of centre of gravity: the self doesn't really exist either. Just like the 'self' the centre of gravity is an abstract concept that amongst

others provides a way of predicting behaviour. A main property of such an abstract concept is indeterminacy. You don't ask questions outside the fictional world of the story. For example, if a character is introduced and nothing in particular is said about him, you don't assume he has a mole on his shoulder and you cannot verify it either. In the real world you can verify this, although you may never get the chance.

Dennet also gives an evolutionary cause for the narrative. He argues that our ancestors started of with utterances conveying questions like "help me". The questions where not directed to anyone in particular and where asked whenever they didn't know how to solve a certain problem. Then one day one of our ancestors stumbled upon a problem, asked a question, but no one was there to answer. Still, by hearing himself asking this question he thought of an answer. This ancestor communicated with another part of his brain through talking to himself. These parts of the brain didn't have access to each other because of some "deep biological reasons".

Talking to oneself still plays a major role; according to Dennet conscious thinking is a form of talking to oneself. Only the route from mouth to ear got shorter changing into conscious verbal thought; vocalization was an inefficient part of the loop.

Dennet doesn't really specify the step made from utterances to narrative. It seems that the complexity of narrative is needed to cope with finding a coherent story within the brain's disunified parts, it is not mentioned if there is a particular structure to it (see section 4.5).

First some overlapping issues with Dennett's work and the previous chapters should be pointed out. After that another scientist (Carr) is discussed who provides us with a possible clarification to the differences between these theories.

Communication

Compared to the previous section (Dautenhahn, section 4.3.2) communication still seems to be a constituting property of narrative. The only difference here is that narrative's primary use seems to be finding internal coherence, constructing a biography; communication was initially in the form of utterances. Communication with other's in narrative format could have arisen from that, but is not mentioned by Dennet.

Language

Another difference with Dautenhahn which Dennett's theory implies is that (some kind of) language had to be evolved before narrative. Dautenhahn's preverbal transactional format suggests that before language already some primitive narrative format existed. This narrative format arose from social interactions between primates.

Coherence

The similarity between Dennet en Dautenhahn's theories is that narrative in both their theories is used for finding coherence. Dautenhahn's theory can be viewed as that narrative is used for finding coherence between individual primates (a group) instead of finding coherence within the brain of an individual. She gives better insight into the format of this narrative while Dennet doesn't.

Carr shows that the gap between Dautenhahn's social perspective on narrative and Dennett's individual perspective doesn't have to be that big [3]. Although he doesn't go into the possible evolutional origins like they do.

Carr starts of with a discussion on the role of narrative within an individual. According to Carr everyone is the subject, teller and audience of its life-story. The problem of self-coherence is unifying these roles.

As mentioned in section 4.2 Carr says that humans can only see the present in the context of past and future, this he calls 'human-reality'. While going through life humans make a story of their past en present, while their future actions are also anticipated in the story. Sometimes the story is changed according to the actual events and sometimes the events are changed according to the story. The individual does this by telling the story to him/herself, being the subject and the audience of it.

There are many similarities in Carr's theory with respect to Dennett's theory. Both say that humans talk to themselves constructing an autobiography with them as the main character in order to find self-coherence. A difference is that while Dennett's theory in this particular paper [10] only covers self-coherence with respect to the individual disunified brain, Carr's theory also covers self-coherence with respect to events outside the human body. However, it should be noted that Dennet focuses on finding a primary, evolutional purpose of narrative.

Carr continues by arguing that his theory on the individual in human-reality can be extended to its specific social form, although he acknowledges that it his ideas where social from the start. Storytelling is a social activity: "… the story of one's life and activity is told as much to others as to oneself" [10: page 18].

In its specific social form, the subject of the story is the group to which the individual belongs; the subject turns from *me into us*. The story is told and made by the members of the group, which all play their individual roles. The story must be believable by the audience in order for the members of the group to act out their role.

Narrative in this way can provide coherence for a group in addition to the self. Carr even extends his notion to community. A story can constitute a community, representing what is happening to them, what they are doing and who they are.

At the end of his essay he suggests that the autobiography could well be a specialized story in respect to the social story; both kinds of story are an interplay of roles. He defends starting at the individual perspective by saying that finding individual coherence is a matter closest to us all. He concludes by saying that both kinds of stories owe their existence to each other if the individual story is at least partly social in origin.

Communication Revised

Looking at Carr's theory, communication is a basic property of narrative. A human can communicate with himself just as well as with others. Also Dennett's theory could be seen as that humans first started to communicate in narrative format with themselves, in order to cope with their disunified brain.

Culture

In section 4.3.1 Sengers idea of culture – derived from Bruner's – was mentioned. She talks about culture being constituted by mini-stories linked into a bigger story. If Carr's notion of community could be extended to culture it would provide better insight into the term culture. Culture doesn't have to be a simple concept, as Carr states [3: page 22] a community can consist of other communities which can even be in conflict with each other.

Non-coherence

While striving for self- (and group) coherence is considered as the most important use of narrative by Dennet and Carr, they both mention that it's not always possible to reach coherence. Dennet gives the example of the multiple personality disorder, this term seems to arise from the fact that the person's autobiographical story doesn't seem to belong to 1 self. Carr talks about a groups "centrifugal tendency to fragment" [3: page 20].

Summary

- Before the existence of narrative our ancestors could access parts of their brain by hearing themselves talk, which they otherwise couldn't access, according to Dennet. Through evolution saying things out loud wasn't necessary anymore; the route got shorter resulting into conscious verbal thought.
- Parts of the brain are essentially disunified; constructing a coherent story provides coherence between those disunified parts.
- Carr clarifies this narrative process more detailed. Humans anticipate on future, present and past by making a narrative with them self as the subject and the audience. Sometimes the story is changed according to the events and sometimes the events are changed according to the story. This anticipating process is discussed in more detail in Bordwell's theory in section 4.5.3.
- Dennet's and Carr's theories are not incompatible with that of Dautenhahn, by changing the subject from *me* into *us*, the same process of finding a coherent story then is performed by the group, providing group coherence.

4.4.1 Knowledge Representation

Topics

- In the previous chapter a philosophical argument was made for the individual purposes of narrative. Using Schank's theory it is explored more precisely which processes are at work here.

In the previous chapter a case is made for narrative as providing individual coherence as well as group coherence. A even stronger case is made by Schank and Abelson [33, 34], who argues that virtually all knowledge is based on stories. Schank's perspective is that of Artificial Intelligence; the focus of his work is on the dynamics and underlying constructs of stories in humans.

The style in which his theory is presented is a practical concrete one, making it suitable for translation to computer science applications but sometimes generalizing a bit too much. In another way the essence of parts of his work can sometimes be obscured by a lot of examples and omitting details about terms like ‘gist.

Schank presents his view on stories in a practical and concrete way, using a lot of examples and often making suitable for translation to computer science applications. His theory is a new one and in combination with the examples the total extent of his theory can be hard to extract. He acknowledges that their style of presentation is “prone to overstatement” in order to make their case.

The main reason Schank is mentioned in this chapter is that he emphasis the use of stories in knowledge representation: “virtually all human knowledge is based on stories constructed around past experiences”. In addition to this his ideas on the construction and forming of these stories will be reviewed. The more concrete aspects of his work on story skeletons and indices *(more?)* will be left to section 4.5.

Schank’s first proposition in [34] is “virtually all human knowledge is based on stories constructed around past experiences”. Schank argues that stories are better than generalizations or abstractions. Stories represent functional knowledge to facilitate daily use; this is opposite to logical inferences of experiences, for which human’s don’t seem to be setup for. Knowledge represented in stories provides a context and allows for reassessing what the story means for the person, possibly in relation with a new contradicting story. For humans it’s hard to remember abstractions, they need to place them in their particular experiences

Schank older theory on scripts [31] is not outdated by his theory on stories. Scripts store knowledge for daily situations; it contains for example rules of how to behave in a restaurant. Scripts allow for learning, they can be adapted through time. In familiar situations scripts can be used, which relieves us from having to ‘think’ about the same situation time and time again. When no scripts seem to apply, stories provide a way of interpreting this new experience by relating them with old experiences.

Schank also calls the knowledge contained in scripts ‘general event knowledge’ or ‘event based memory’. It is stored in a certain place in memory that is updated every time something knew I learned about the particular subject. This way events of the particular day are forgotten, the knowledge is generalized for a certain subject. A place in memory can be devoted to the local grocery store, where the milk is located and whether or not to pay with credit card is contained in this place. This kind of memory is still episodic rather than semantic because it is based on actual experience. The repetition of daily events destroys the coherence between particular events, to remember them as a whole they should be put in sequence (trying to find coherence) by telling them. Although stories can be told in many different ways, different versions of one story are often so similar that they must be stored and retrieved as chunks.

According to Schank stories consist of experiences/events in combination with an index. Indexes are generated from new stories by relating them to old stories. The more indexes are created for a story, the better we can relate it to new incoming

stories, the better our learning. This is much in accordance with Schacter [30: page 39-71] , who shows that encoding of experience has much influence on its retrieval. A person's subjective perception of this experience plays an important role, which could be related to Schank's theory in the way that old stories influence this perception. This process reflects Schank's second proposition: "new experiences are interpreted in terms of old stories".

Schank's idea about the dynamics of a story can be best described by his view on conversations. A conversation is a series of remindings. When a story is told in a certain conversation, this makes one remember another story. This story then is adapted to make it more suitable for the conversation after which it is told. Adapting the story and telling it in turn changes the tellers' memory. A rather strong statement that Schank makes is that all stories one ever is going to tell are already made up. He says that most of the time old stories are reformulated; the *gist* is re-expressed (for more about gist see section 4.5. Also he argues that when you find a close matching story this signals you have understood.

The stories in a conversation are already present in memory, but what about new experiences? By formulating a new story certain details of an event are selected and others are filtered out. According to Schank the formulation of a new story forms memory. A story has to be retold to remember it, by retelling more details can disappear from the story and embellishments can be added. The story can also become a fragment; the big picture is not remembered anymore.

It must be noted that new stories about new experiences are still interpreted in terms of old stories. In this, a greater familiarity with the new situation induces a greater reliance on old stories. The way old stories fit to new stories influences interpretation of the new story. Furthermore stories tend to be interpreted in terms of the personality characteristics of the main actor. Schank uses research from the field of psychology to show that there are some other factors influencing which old stories come to mind first when interpreting the new story [34: page 31-33].

Considering the formulation of stories as constituting memory implies two things. When false stories are constructed and told by someone, there is a risk that after a while this person doesn't know they are false anymore. Another implication is that *not telling* stories can cause someone to forget.

Behind the Story

It is hard to find out what Schank's ideas are about whether or not a story is actually made, because he doesn't explain this explicitly. As said, daily and often repeated events end up generalized, others are put into a story by *telling* them. It is than safe to assume that the appropriateness of the story in the social environment of the teller heavily influences whether or not a story is told, and thus remembered.

Whether or not people tell a story depends on if this story is unusual (for the audience) [33: page 36]. It should be unique and have personal significance [34: page 37]. If it is not constructed the story will disappear from memory.

As said before, negative experiences can be told to oneself, still remembering what happened. This, and Schank mentioning 'personal significance' suggests that there are

other ‘drives’ at work deciding what is made into a story. It could depend upon the extent to which an experience relates to another story in memory, but Schank also mentions that ‘anomalous’ events can be stored into a story [34: page 19].

A related issue is the issue that Schank puts forward about what comes first, the belief or the story [34: page 13]. He argues that it makes no difference in the end and that the belief ends up as an index of the story. This issue is related because it raises the idea that a story can influence for example ‘personal significance’. On the other hand, a belief seems to be a particularly complicated (higher) concept.

It could be possible that the notion of an underlying mechanism independent from stories is needed here. Though, due to time limitations this will be left for further research.

Boundaries

Schank’s talks about event-based and story-based memory, arguing that both are episodic in nature. He denies the existence of semantic memory; event-based memory is generalized but still depends upon actual experiences.

Trying to defend his statement that virtually all human-knowledge is based on stories Schank argues that facts don’t exist in memory by itself. Searching in memory is searching for a story and facts are derived from stories.

He doesn’t say knowledge of words, numbers, grammar, rule systems etc. are fundamentally based on stories (or episodes). A sentence is not a combination of stories and a theorist can prove a theorem and words can be learned by rote.
Their strong argument that virtually all human-knowledge is based on stories seems not entirely proven, but they are trying to show facts are not unrelated from stories. They do this by showing that numbers and words (or utterances) for example can be an extraction of a story and in turn refer to a story as an index.

He argues that in order to become useful in a social and cognitive way they must be contained in a story. Stories make the ‘factual’ knowledge available in social communication and make the knowledge be remembered through time. Stories can be richly indexed making them ready for the interpretation of a high variety of circumstances. Without a story Schank cannot imagine a useful conversation.

Still, the argument is not entirely satisfying. Several scientists argue there is more than story related knowledge [41], like imagery and music. Schank dismisses those arguments by saying that those alternative forms of knowledge have either real stories behind them or by stating that this alternative form isn’t really knowledge.

> *Schank’s last statement, “the content of the story memories depends on whether and how they are told to others, and these reconstituted memories form the basis of the individuals remembered self.” is much in relation with the introduction of this chapter and the previous chapters. This statement is discussed on the basis of several common aspects:*

Individual Coherence

As already said, in order to remember individual events to have to be put together into a story in a coherent way, this way coherency is found between events. However, this can be taken a step further.

According to Schank we define ourselves by telling stories, one decides in which form to tell the story depending on which view one has of oneself. The stories constructed and told change ones self-definition because what we tell is what we remember. Becoming an adult means knowing what your stories are, you have to think less about answers when a question is asked because you have your stories ready. This relates much the previous arguments on stories used for finding self-coherence.

Group Coherence

Schank also gives importance to narratives in a social, group context. People have to be in close-relationship and have to share many experiences. When their individual versions of the stories are close they have found a "common ground". In a group like that utterances can be enough to refer to a shared story. Referring to the introduction of this chapter this could be seen as finding group-coherence through stories. Schank call's these stories 'shared story memory' or co-biographical memory, in addition to the autobiography identified earlier.

Culture

The former sub-chapters about coherence are more or less interpreted in terms of the introduction of this section (4.4), but relate much to Schank's view on culture [33: page 189-218]. Schank says people need to learn stories of their culture, as well as adopt stories of their own. In this, being part of a (sub) culture means that you talk in terms of its stories while being independent means relying more on your own stories; growing up means learning both kinds of stories.

Schank talks about cultures and subcultures, a group belonging to one culture can be of any size. It seems that Schank sees 'culture' as a country's culture, while a subculture can be a certain movement in society down to a group of friends. People live in multiple subcultures at once.

Cultures and subcultures basically consist of three types of stories: neutral, condensed and elaborated stories. Neutral stories are very generalized and seem to apply to almost any situation. Condensed stories are generalizations of many stories; proverbs are a good example of them. Condensed stories are helpful in being creative; they can be fitted to new situation. Parts of the condensed story can be filled in according to the new situation in unorthodox ways to achieve 'creative' results.

Elaborated stories are like condensed stories, but are more particular and have more details. Myths and heroic stories are an example of these. Elaborated stories are used the other way around; people use these stories as a guideline, adjusting their life to the story instead of adjusting the story to the situation (for example 'songlines' in the Introduction). Elaborated stories are the ones playing a role in finding group coherence; they create a bond between people because they can refer to them.

As Schank's argues, all we know is contained in stories and we understand things in terms of our stories. 'Original' thinking would be the adaptation of stories of ones culture. Also culture supplies us with standard explanation patterns, these stories are generally and easily accepted in the culture.

Stories are not so easily adopted as one may think. A great teller tries to relate his stories as close as possible to the listener's situation. It is important to make the story personal and particular, trying to relate as much as possible to personal experiences. Though people want to define themselves through other people's stories, they tend to find their own similar story and reinforce it.

Schank's theory fits in with the aspects of culture identified in the previous chapters. Both Dautenhahn and Schank talk about the influence of culture on story format. Schank's neutral and condensed stories and his notion of explanation patterns can be an example of Dautenhahn's cultural canonical forms of interaction. These stories and the elaborated stories form the stories of ones culture, in analogy with Sengers.

According to Sengers narratives forming a culture are linked. As both Schank and Carr (more freely interpreted) say, culture can consist of subcultures, which gives rise to conflicts between stories. Sengers doesn't think this has to be a problem, even if narratives of the same (sub) culture contradict. Stories from a culture or subculture could be linked by their common theme. This could be analogous to stories that define oneself (autobiographies); the common theme in these stories is the view that one has of oneself.

Communication

In Schank's theory stories constitute ways for storing virtually all that humans know and provides them with intelligence (see section 4.4.2). While in this chapter mainly internal uses of story are of importance, communication seems inseparable from stories. Schank acknowledges that it is still an open question why we want to tell stories [33: page 234]. As we will later discuss in section 4.4.2, telling a story helps one notice new things and find the essence of the story.

Communication also seems to be essential for memory. As already said, events have to be put together in stories by telling them in order to remember them, and without a story Schank cannot imagine a useful conversation. The other way around he argues that accepting a story told by someone is remembering it.

Telling to Oneself

As said in the previous in *Communication,* Schank thinks telling a story helps one notice new things. The fact that someone hearing himself talk helps this person find out new things shows similarities with Dennet's theory (see introduction of section 4.4). Dennet argues that the internal separation of parts of the brain prevented ancestors of humans to find answers internally, but by telling them they could connect the necessary brain parts. This could be a good explanation why people need to tell stories.

So, in essence, the benefit of telling a story was not a social one, it had a pure individual purpose. Dennet even argues that the brain adapted so that stories (or

utterances) didn't have to be told out loud anymore. Schank nuances this a bit more, arguing that actually telling it out loud (to someone) is more beneficial.

Schank's view on negative experiences [33: page 141] reveals more about telling versus not-telling stories (to others). Negative experiences are sometimes formulated in a story but not told, they are only told to oneself. This way incoherent stories are allowed to exist, we don't examine the story for inconsistencies. Schank argues that telling a story, to another person in particular, helps making the story more coherent.

Language

In Schank's view language seems to be a prerequisite for communication in stories, stories have to be formulated in a language. This is in contrast with Dautenhahn's preverbal transactional format of narrative, which suggests language is not necessarily needed. It seems quite plausible that narrative can exist without verbal language; narrative can be communicated through body gestures, images and movies. On the other hand, these can be seen as a kind of language.

Reality

Schank makes a difference between stories and 'reality', assuming that stories distort reality. Forming events into narrative leaves out certain aspects and finds structure (coherence) where there is none. Schank assumes the existence of an objective reality [34: page 52], story skeletons (discussed in section 4.5) seem to help humans create their 'own version of truth'. Also it seems that according to Schank our brains prefer disconnecting (daily) events from those that follow [34: page 40].

This view corresponds with the 'common-sense' view Carr identifies [3]. As already mentioned Carr argues against this discontinuity between narrative and reality. Most important is Schank's implicit notion of reality; according to Carr there is no 'objective reality' but only human reality. Narrative is an extension of one of the primary features of the events it relates. For more about Carr's theory see section 4.2 and the introduction of this chapter (4.4).

Summary

- Schank makes two propositions
  - Virtually all human knowledge is based on stories
  - New stories are interpreted in terms of old stories
- Humans have trouble with abstractions; stories represent functional knowledge for daily use.
- Apart from stories there is ‘event based knowledge’, used for repeated daily activities like ‘going to the grocery’.
- Stories consist of events and an index.
- A story has to be told in order to remember it, telling a story to oneself doesn’t remove inconsistencies as well as telling it to another person.
- In relation with coherence:
  - Stories are told depending on the view that one has of oneself, these told stories influence what is remembered of the story. This way individual coherence is reached.
  - When people share many experiences their stories are very similar and can be referenced by a mere utterance. This constitutes the former mentioned group coherence.
- Schank identifies 3 types of stories in culture: neutral, condensed and elaborated stories. These stories form a base for ones individual stories.
- Schank focuses on language but doesn’t mention body gestures (Dautenhahn), images and movies.
- Schank assumes an objective reality; according to Carr a human can only experience ‘human reality’.

4.4.2 Intelligence

Topics

- In the previous chapter it is argued that knowledge is represented and used in the form of narrative, here the question is answered how this can result in intelligent behaviour.

After having discussed all previous aspects, the most important question remains, how does behaving intelligent stem from using narratives? This doesn’t seem very straightforward, first of all what does being intelligent mean?

The Oxford Encyclopedic English Dictionary:

intelligent
1 having or showing intelligence, esp. of a high level
2 quick of mind; clever
3
- **a** (of a device or machine) able to vary its behaviour in response to varying situations and requirements and past experience
- **b** **…**

intelligence
1
- **a** the intellect; the understanding
- **b** quickness of understanding; wisdom

2 **…**

intellect
1
- **a** the faculty of reasoning, knowing, and thinking, as distinct from feeling
- **b** the understanding or mental powers …

2 **…**

understand
1 perceive the meaning of …
2 perceive the significance or explanation or cause of …
3 be sympathetically aware of the character or nature of, know how to deal with …
4
- **a** … infer esp. from information received, take as implied, take for granted
- **b** … believe or assume from knowledge or inference

5 **…**

Chambers Concise Dictionary:

intelligent
endowed with the faculty of reason: alert, bright, quick of mind: well-informed: cognisant: capable of performing some of the functions of a computer (*automation*).

Intelligence is obviously a term that covers a lot of aspects, which is a study in itself to explore. However, in this chapter intelligence is seen with respect to stories, from this view intelligence will be defined more specific and possibly different. Though, it is useful to have a look at the properties typically assigned to the word intelligence.

Looking at the above extractions from two dictionaries understanding, reasoning, thinking, and possessing knowledge seem to be basic properties. On top of that, quickness and alertness are also valued as important properties. Understanding is a particular complex concept, which concerns how things are interpreted. It involves extracting meaning, significance and drawing conclusions.

Having knowledge is obviously important for being intelligent, what is to be understood if there is no information to understand? In the previous chapter some benefits are mentioned why narrative is particularly useful for carrying knowledge, of which being able to re-interpret it and its daily applicability are the most important.

Another interesting thing is that intelligence is seen as being separate from 'feeling'; being able to perform functions of a computer. When looking up definitions of 'thought' and 'reason', the same trend can be seen; both concepts seem to involve more 'clean' and abstract thinking.

It's here where the usage of narrative will shed a different light on intelligence. In Dautenhahn's argument intelligence originates from the necessity of maintaining group coherence, which was done by social interaction. Because this social interaction became quite complex in bigger groups, the brain evolved to cope with this.

In the start this social interaction came down to physical contact, grooming. Verbal language made this social interaction more efficient, the narrative format of language came into existence because it is suitable for conveying the same information as the original grooming did. The narrative format even extended the communicational capabilities of grooming because it could include fictional persons or persons who where not there. According to Dautenhahn narrative can also provide sensual, emotional and meaningful aspects, enabling subtle nuances.

Because narrative allowed for even more sophisticated interaction, the brain could evolve further. This way, narrative can be seen as a very versatile tool for social interaction, it created the conditions for higher intelligence.

Then, if narrative is basically social, intelligence could be much closer to feelings and 'being human'. Intelligence possibly isn't clean abstract thinking; it could be a lot messier and speculative.

Schank's seems to have the same view: intelligence is not "a now you have it, now you don't affair". Intelligence depends on the extent to which certain capabilities are used. Most important here is that according to Schank the human mind could work "inherently simpler than AI researchers have wanted to admit", which can be extended to the term intelligence.

Dautenhahn and Schank differ in the fact that they focus on different aspects of intelligence. While Dautenhahn looks more at the origin and the (preverbal) structure of narrative, Schank focuses more on how narrative is used and stored. As the chapter layout suggests, Schank concentrates more on individual aspects of narrative while Dautenhahn views it as being basically social.

Both Schank and Dautenhahn focus on narrative as a tool of finding group coherence. While in Schank's theory 'telling' stories is very important, individual coherence is also an important aspect, which is also in accordance with Carr's theory.

Schank identifies a series of capabilities (in terms of stories) which establish intelligence [33: page 219-243], most of these are already mentioned to a certain extent in the previous chapter. For each capability he specifies a basic utilization of

this capability and an extended one, marking higher intelligence. Both are mentioned explicitly for each capability (first=basic, second=extended):

- **data finding:** *getting reminded, searching for data*

  Getting reminded of something is a natural process while searching for data isn't and is learned. Both depend almost entirely on how stories are labelled (indexed); what you didn't label you can't find back. In order for labelling to be optimal, stories need to be labelled in a careful, consistent and complete way. Events need to be related to other stories as much as possible, old stories have an influence on the labelling of new events. Having more interests helps labelling stories in a more complete way.

  Intelligence also involves forgetting irrelevant stories, focussing on the significant. Higher intelligence involves 'mulling' over stories, as Schank calls it, which helps generate more labels for stories. Furthermore it depends on consistent labelling, which helps finding less obvious cases in certain situations. The question here is if Schank's notion of mulling is somehow similar to talking to oneself.

- **data manipulation:** *partial matching, adapt old data*

  New events are matched to stories that are not exactly like those stories. Higher intelligence involves keeping unlikely matches alive; there are no pre-defined rules for this. Adaptation of old stories to fit new events is an important activity in this. However, no match will almost ever be a complete match; the validity of a match can only be found through trial-and-error.

- **comprehension:** *connect new stories to old stories, invent coherency*

  New stories are being related to older stories in less than obvious ways, often used to explain the behaviour of people. Higher intelligence means looking more intensively for coherence or even inventing coherence where there is none.

- **explanation:** *explain expectation failures, discover predictive rules*

  Stories can be seen in a variety of ways, this way they can explain a certain situation in different ways. What is the right way can only by found out by means of failure; when a certain 'explanation' doesn't turn out right, the story has to be reinterpreted. A set of predictive rules of how a certain event will turn out can be extracted from a story. The original story should still be available though; in case that the rule fails the story has to be reinterpreted.

- **planning:** *execute plans, create plans*

  Schank argues that humans and even animals execute plans, conscious or unconscious. Plans are copied from family, friends etc., recalling the right plan at the right time comes down to the capability of *data planning* (i.e. labelling) as described earlier. Humans prefer adapting old plans in new situations, rather than creating new plans, which is a more advanced characteristic of intelligence.

- **communication:** *tell stories, generalize/crystallize/elaborate stories*

  Communication enables us to discover new things in a story, crystallize it and find its essence. In this, hearing ourselves talk helps, but also stories other people say in reply can help us with that.

  A good storyteller knows how to tell the story to relate best to its audience and the current topic, maximizing the communicational benefit on both sides. Less intelligent tellers tell all details, leaving out the point of why they are telling it. Higher intelligence is marked by the ability to elaborate on old stories, coming up with (entertaining) new stories that don't necessarily have to be true.

  Finally, Schank says that people are not born as fascinating storytellers; storytelling is learned by caring about improving and paying attention to ones audience.

- **integration:** *understand stories that we have been told, being curious*

  The problem here is what to do if a certain story doesn't fit into other stories. Integration is about deciding what stories to integrate in our old stories (Schank talks about world model) and what not, it is not possible to remember everything. 'Smart' people ignore what they don't need to know or pay less attention to it.

  Higher intelligence means getting curious about certain things and wanting to learn more about them. This makes intelligence concerning this capability a subjective thing. Having knowledge about some issues can be more fashionable in society than other things, colouring the view of others whether or not one is 'intelligent'.

Summarizing, Schank adds the concept of indexes (labels) to stories. His whole view of intelligence depends upon this labelling process; one cannot behave intelligent when one doesn't organize his stories well enough to find them back when needed.

Being able to look in different ways at the stories one possesses in order to make them applicable to a particular situation is the basic process needed to use stories in a useful way. This way a story can give a certain 'meaning', 'explanation' etc. to a situation, making this process very similar to the description of 'understanding' from the Oxford dictionary.

The capabilities of manipulation, comprehension, explanation, integration and even planning are all the same in this way. Also labelling involves the same process, in order to perform meaningful and complete labelling it is good to look at the story in various different ways. What the 'right' way to look at a story is can only be found out by means of trial-and-error.

Communication is a way to enhance the process of being creative with stories. Telling those helps thinking about them, finding its essence; also other people's stories can provide new insights.

The hallmarks of intelligence, as Schank calls them, often involve going beyond the boundaries of the stories one has. Manipulate stories in unlikely ways, finding coherence where there is none and elaborating stories beyond ‘truth’. Being very loose in interpretation constitutes creativity.

‘Dull’ people keep with the stories they get and they don’t spend a lot of time at shaping their set of stories. They also don’t filter out information in the most efficient way. ‘Smart’ people are better at selecting knowledge, and they seem to be less a subject of a fate. A key difference here seems to be that smart people have more interests which steers what they will remember and helps labelling the stories better, more coherent and complete. This, in turn, helps them come up with more useful stories at the right time.

According to the two dictionaries cited above, quickness and alertness also are qualities someone intelligent shows. Here complete and consistent labelling would enhance correctness and quickness of recall. Also knowing what not to pay too much attention to (or forget) would speed up ones reaction time. One would not have to think about everything, leaving more time for processing relevant stories and coming up with relevant ones.

Considering the above, intelligence doesn’t seem to come plainly from stories, but provides the basic framework for it. The main process of manipulating stories (either by telling them, or internally) doesn’t seem to be a thing that just ‘arises’. But the way stories are structured and more insight in its content could point out how trivial or non-trivial the act of manipulating stories actually is, which will be looked upon in the next section (4.5).

From Schank’s work it can be seen that having interests is very important as a guide through the mass of information a human goes through each day. How do these interests come about? It could be a clue to something behind the content of stories, due to time limitations this will be left for further research.

Summary

- Understanding and possessing knowledge are important properties of intelligence, but also quickness and alertness are often associated with intelligence.
- In the dictionary the word ‘intelligence’ and related words are often related to clean and abstract thinking, like a computer. Both Schank’s and Dautenhahn’s theory suggest that intelligence may be inherently simpler and not distinct from ‘being human’ (for example feelings and social behaviour).
- Schank defines a series of story-related capabilities that one needs in order to behave intelligent.
- The better one can label story-based knowledge in a complete and consistent way the more intelligent one is.
- Thinking about and telling stories helps in labeling stories better.
- Having interests is important for filtering out ‘relevant’ information, resulting in more consistent labeling and quicker recalling of stories.
- ‘Higher’ intelligence’ means that one can manipulate and interpret stories in more unlikely ways.

4.5 What is the story?

Topic

- More concrete aspects of Schank’s and Bordwell’s theories are globally discussed. The similarities and differences between their theories will be pointed out.

The main influences for this chapter are the work of Bordwell [2] and that of Schank [33, 34], because of their applicability to the subject of this thesis and their quality. Their works fill add to each other quite well; were Schank is not specific enough, Bordwell is, and vice versa.

Bordwell’s theory on narrative stems from film analyses, but its significant part is defined in a medium-independent way, which could prove very useful in the application domain of this thesis, as there doesn’t have to be a restriction to a certain language. This is in contrary to Schank, who often refers to the English language as a primary means of representation for narrative. According to Bordwell, language is “…an instrument of and guide for mental activity.” [2: page 30].

Most important in Bordwell’s work is the separation between the *Fabula*, *Syuzhet* and Style of the narrative. Basically the Fabula is the representation of the story as it is constructed in ones head, the syuzhet is how the story is presented to the perceiver. This will be discussed more elaborately in section 4.5.2. The separation between Fabula and Syuzhet originates from Bordwell’s constructivist point of view. This view basically means that narrative doesn’t comprise only the structure in which it is presented (syuzhet), but that the viewer dynamically constructs the narrative model (fabula) from ‘cues’ given in the syuzhet.

The way the narrative is constructed in humans depends on perceptional characteristics, for example that 24 frames per second in a movie are perceived as continuous light. Also cognitive aspects influence how the narrative is constructed, like knowledge and expectations. Bordwell doesn't make a separation between the two in his theory. He mentions that it is important to realize that the perceiver doesn't have to be an ideal one; a human perceiver can be tired and miss cues given by the syuzhet, which influences the construction of the fabula. This could be an advantage an artificial perceiver has over a human perceiver.

As a third dimension of narrative, Bordwell mentions Style. In a movie this could for example be different ways of zooming in and the camera angles in which scenes and people are shot. Because style is more medium-dependent and doesn't seem very relevant to the subject of this thesis, this part of his theory is not discussed here.

In this chapter the structure of narrative is discussed, which makes it seem inappropriate to include a discussion of the seemingly subjective way of constructing the fabula. Still, the processes and memory structures (called schemata by Bordwell) involving construction of the fabula are specified specifically by Bordwell. He acknowledges that to certain extent these structures make construction of a narrative subjective. Though, he argues, "…in principle, viewers of a film will agree about either what the story is or what factors obscure or render ambiguous the adequate construction of the story." [2: page 49].

He says that the perceiver's comprehension of a narrative is theoretically separable from his or her emotional responses. Comprehending and recalling a story is remarkably similar for all age groups and people share a sense of what is secondary to a stories point and what's essential. It is interesting that Schank is more sceptical about this: Schank says that often people don't use the same story skeletons, and that people create their own version of the truth. This version can be a distortion of reality in which one can even start believing. A reason could be that Bordwell has pre-made stories (movies) in mind, while Schank mainly talks about real-world experience. This suggests that there could be other more subjective influences at play here, as said in the introduction of chapter 4 this will be left for further research.

Bordwell's separation of fabula and syuzhet are used as a basic guideline for discussing the structure of narrative. Looking at Schank's theory, it is problematic to combine his theory with Bordwell's. The main difference here is that Schank's view is more an information-processing model based on his older work on Scripts, Plans and Goals [31], and Case-Based Reasoning [32]. Also contradictions within and between his books on storytelling [33, 34] make it hard to lay a finger on how a story is actually constructed in memory.

In both his books he talks about events and experiences to be included in this constructed story, but the more concrete parts of his theory seem to boil down to storing a generalization of the story instead of storing its specific events. In his first book [33] he discussed index and gist separately. His description of story indices is based on his work on Script Plans and Goals [31]. In this description he says that "memory structures and processing structures [have] to be one and the same in order

for reminding to take place", by which he means that the index is actually everything that is stored from a story.

While in a later refinement of this book [34] he still says that stories consist of experiences and labels (indices), but he actually doesn't use the term gist anymore and literally replaces the word gist with index.

Schank neglects to give a notion of how the specific experiences (events) are included in this gist while, as included in section 4.4.1, he actually argues that specific experiences are needed in order to be able to reassess a certain rule (included in the index) extracted from this story. This is the reason that Bordwell's theory on the fabula is used as a guideline for how specific events are stored in memory, as discussed in section 4.5.2. Schank's index (or gist) is used as a concept for indexing this fabula, which will be looked at more closely in section 4.5.1. For this reason his older notion of index is used from [33]. It is not certain if he still sees indices to be like this, as he doesn't mention it in a later refinement of his theory [34] and actually talks about events being put in the index.

Although Schank's index is viewed as being more general than Bordwell's fabula, they are not entirely incompatible as will be shown in section 4.5.4; also some concepts in the two accounts are very similar to each other. As will be seen Schank's story skeletons are very similar to Bordwell's template schemata, just as the way both define how stories are manipulated (section 4.5.3).

Bordwell's concept of syuzhet doesn't seem to apply directly to the subject of this thesis. The syuzhet of a story is sometimes created to challenge the perceiver of the story; construction of a coherent story is often made difficult by different techniques. While this is mainly done for entertainment in pre-written stories, possibly some of these difficulties will arise when one is constructing a story from real-world events. One has to be careful not to interpret his theory too freely. However, while it is not sure if narrative 'cues' can be found in the real world, he doesn't assume that there is a sender of the narrative. In the last section (4.5.5) it will be seen how far this can be taken.

Summary

- Bordwell's theory is medium-independent, Schank's focuses more on the English language.
- Bordwell's separates between Fabula and Syuzhet.
  - Fabula: story constructed in the mind
  - Syuzhet: story as it is presented to the perceiver
- Bordwell's theory is more about interpreting pre-made stories, while Schank's is about constructing stories (fabula) from real-world experiences.
- According Bordwell different people construct a more or less similar fabula from the same syuzhet, Schank's theory is more sceptical about this: people create their own version of the truth.
- Schank's stories (fabula) tend to be generalizations, while Bordwell's fabula contains specific events. Because Schank's stories are more general they can function as indexes.
- Schank's story skeletons are very similar to Bordwell's template schemata (as will be seen later).

4.5.1 Index

Topic

- Detailed discussion of Schank's *index*.

In his first work on stories [33] Schank provides a quite concrete definition of what an index might be. This definition finds its origin in his earlier work on Case-Based Reasoning (CBR) [32] and Script, Plans and Goals [31] but is extended to the domain of storytelling. Though his work on indices and CBR is a research field in its own right, a general outline is given here in relation to stories.

In section 4.4.2 the importance of indexes, or labels, is pointed out. Stories need to indexed and labelled in a careful, consistent and complete way; a story can have multiple indexes. The right story needs to be found at the right time.

According to Schank an index is composed of the same elements of the story that were used to understand the story. This way, the index is not only a structure for retrieving stories but represents a standard vocabulary in which stories is understood. Using this standard vocabulary will generate a consistent indexing, which is an important feature of intelligence as mentioned in section 4.4.2.

He acknowledges that different ways of labelling are possible and that nobody teaches us how to index (and thus understand) stories. Humans create their own way of seeing the world using some generally accepted parameters. He says that our indexing schemes do not differ in principle.

Also he says that the story is stored in the same way as they are indexed and understood, but as already pointed out, Schank's definition of the index is a

generalization leaving out specific events. It was already argued that this is insufficient.

Schank starts of with short generalized stories, mostly consisting of one sentence, representing the basic understanding of the story; this generalization is needed as a starting point of indexing. Though this generalization can be seen as an index, he argues that a way of indexing is needed which is independent of the words to express them. As a starting point he uses the indexing language as defined in his work on Scripts, Plans and Goals. The generalized story (sentence) is broken down into three elements:

- **Topic:** general topic, often about personal preoccupations / problems.
- **Goal:** goal generated by, and related to theme.
- **Plan:** generated plan to achieve goal, gives rise to standard courses of action.

To this he adds two additional elements:

- **Result:** intended or unintended result of pursuing goal.
- **Observation:** new belief or point of view, which was the point of the generalized sentence, specified abstract and domain independent.

For example, Schank mentions the following generalized sentence [33: page 108]:

"In deciding whether to enter a long-term relationship with somebody, assume that the behaviour you dislike in the person will continue to manifest itself, and make sure that you are willing to live with that behaviour".

Schank (personally) would construct the following index of this sentence:

- **Topic:** establish long-term relationship with another person for a purpose
- **Goal**: find employee
- **Plan**: choose person whom you like but who exhibits a bad behaviour
- **Result**: bad behaviour causes new problems
- **Observation**: Learn to read the signs better

It should be noted that he initially defined *topic* and *observation* as *theme* and *lesson*. Because the elements topic and observation constitute a more general specification of an index, these are used in this thesis. Theme and lesson relate to specific advisory stories, but in general stories don't give an absolute truth, which is presumably one of the major strengths of a story.

Of these elements topic and observation are more complex ones, and play a bigger role in indexing than the others. The topic indexes the observation, while the observation indexes the story. The elements *goal*, *plan* and *result* are the actual story; they are contained in the index, as Schank argues that the story <u>is</u> the index.

The story-based understanding process begins with determining the topic. This topic is implicitly discussed, and the chosen topic seems to be more or less dependent on which other topics are common in the perceiver's life. Without an observation it is

hard to remember a story, because it provides an abstraction – the same observations reoccur in different stories. While these stories can seem different on the surface they can still be matched. When the same observation comes up frequently it can become a story skeleton, as the observation already is a generalization of the story contained in the elements goal, plan and result. Note that using this kind of skeleton realizes Schank's idea of interpreting new stories in terms of old ones.

While the topic and observation are abstract descriptions by nature, Schank's examples of goal, plans and results suggest that these are also quite abstract. In order for them to be usable as indices, they require at least some level of abstraction. Schank acknowledges different levels of abstraction are possible for the same story.

Schank doesn't specify how and if an observation is derived from the goal, plan and result. Just as he doesn't define how a skeleton can be used to construct the goal, plan and result, as this is the actual story according to him. Also he sometimes uses the term observation and lesson for the initial generalization that he uses as a start off for constructing the index. This gives rise to the idea that the elements goal, plan and result barely contribute any specifics to the index.

Summary

- An index represents the way in which a story is understood.
- A story can have multiple indexes.
- According to Schank an index consists of the elements: *topic, goal, plan, result* and *observation*. The elements topic and observation play a bigger role in indexing than the other elements; the elements goal, plan and results represent the actual story.

4.5.2 Fabula

Topic

- Constructs related with the construction of Bordwell's fabula are discussed. Analogies with constructs from Schank's theory are given.

Basic ingredients for constructing the fabula are template schemata. This template represents a basic story structure to which new events (or narrative cues) from the syuzhet are fit in order to construct the fabula.

As said, Schank's story skeletons and Bordwell's templates seem very similar concepts though there is confusion whether or not they are used to construct similar structures. In this chapter we will assume they are the same but in order to have a wider perspective on Bordwell's template schemata we will first point out some differences between them. As the similarity between skeleton and schemata already suggests, parts of the index and fabula also show similarities, this is further discussed in section 4.5.4.

First of all, Schank talks mainly about fitting new events (lived experiences) to a skeleton while Bordwell focuses more on fitting events (or 'narrative cues') from a pre-made narrative (the syuzhet).

Schank's skeletons seem to be more complex from the start than Bordwell's templates and are chosen to make the point the teller wants to convey. By choosing a skeleton the teller/maker of the story decides in what way he wants to view the new events, it seems that the nature of the actual events have little role to play in this selection. Again, a reason to believe that more research is needed for this (see introduction of chapter 4).

Also in Bordwell's theory a certain template is chosen from the start which influences further interpretation of the story. While hearing the story, this template (or story by then) is altered dynamically by using procedural schemata when the template doesn't fit the story anymore. Schank defines some complementary processes; both are discussed in section 4.5.3.

Both Schank and Bordwell say that a skeleton / template originates from culture. In Schank's view it seems that there can exist a great diversity of types of skeletons in a culture, stemming from (or being) neutral, elaborated and condensed stories as mentioned in section 4.4.1. Schank often refers to the 'betrayal' skeleton in a divorce story.

These skeletons are generally accepted as valid explanations for certain 'happenings' in these cultures. Schank takes this even further by saying that communication is hardly possible without using skeletons.

Bordwell doesn't give rise to the thought that there is a great variety of skeletons or templates; it seems that a culture has a certain master schema. While this could prove to be a limited view, it provides a good way of further examining this phenomenon in a more concrete way. Bordwell argues that basic structures are needed to interpret stories, just like a painting can be interpreted using circles, squares, triangles etc. Bordwell mentions a canonical story format, often used in western culture. This story format is commonly accepted according to Bordwell and, for example, very similar to Dautenhahn's (better: Bruner's) preverbal narrative format (see section 4.5.6). The latter could mean that this format isn't necessarily specific to western culture; a possibility Bordwell doesn't want to dismiss either, though he acknowledges the matter needs more study.

Bordwell's defines a master schema, which is divided in the following stages:

- **Introduction**: setting, characters
- **Explanation of a state of affairs**: results in goal
- **Attempts**: complicating action, ensuing events
- **Outcome**
- **Resolution**

In this master schema it is assumed that a goal is stated early, which permits the perceiver to fill in causal and temporal connections more exactly. A protagonist is often pursuing this goal.

This master schema is an example of a possible template schema. A template schema in Bordwell's view has to be seen as a filing system. Though Bordwell doesn't specify the exact structure of a template he reveals some characteristics. Note that a template is not a rigid structure; it is subject to alterations by procedural schemata as will be pointed out later on.

The template influences how a story is interpreted. Bordwell doesn't explicitly state if a template schema represents a certain genre, though, the chosen template has a major influence on the rest of the interpretation of the story. A template schema embodies expectations about how to classify events and relate parts to the whole. Events in the template should be discriminable, and be in chronological order and linear causality. Especially the causal connection of events is important according to Bordwell. The template schema defines, as Bordwell eloquently says, "What is likely to be causally prominent" [2: page 35].

Bordwell also discussed the concepts of time and space in relation with syuzhet and fabula. With a small cue in the syuzhet any amount of space can be suggested for the fabula. For example, with the sentence "he travelled all over the world", the world will be part of the fabula. In the context of this thesis the concept of space is of less importance.

In relation to the concept of time, Bordwell says events in the fabula can have both a simultaneous or successive relation. And, in conformance with Schank's view on skeletons Bordwell says that template schemata cause (causal) gaps of the story to be filled in and other parts to be left out or adjusted; a story can get more normal than it was.

Just as template schemata provide a characterization of the overall structure of the story, prototype schemata provide this for smaller components like agents, actions, goals and locales. These descriptions seem to be somewhat stereotypical and can also be images. In Schank's view these schemata are probably stories too, a stereotype story can be seen as a reference to a more elaborate story one knows. To be able to use this stereotype story by telling it, other people have to know this stereotype too, which could mean that this prototype schemata is actually a cultural common story (neutral, condensed, elaborated). It's not sure how the images can be explained by his theory, though it can be argued that an image without an accompanying story cannot contribute to another story (it isn't knowledge).

The prototype schemata and template schemata provide standard expectations for certain events that can be applied by procedural schemata to construct the fabula 'on the fly'. A characterization of this procedure will be given in section 4.5.3.

Though index (or gist in Schank's older work) and fabula are considered as different things in this thesis, both are similar in the way that a skeleton or template is not a prerequisite for constructing them. The resulting fabula / gist depends on how the story finally is manipulated, which doesn't necessarily have to be using templates / skeletons. In the case of Bordwell's theory, the actual story can differ a lot from the canonical story format, and Schank acknowledges the option that no skeleton is used to construct the index.

Procedural schemata, as already mentioned before, can be used for dynamically adapting the fabula. Though, this seems to be a more general structure in Bordwell's theory and can also describe other procedures, like riding a bike. These schemata then, are very similar to Schank scripts, which he also calls 'general event knowledge'. This way Bordwell theory gives rise to the interesting view that Schank's scripts actually could help construct story-based knowledge.

Summary

- For constructing a fabula, template schemata are used. A template schema is a basic story structure to which new events are fit.
- Schank's story skeletons are similar to Bordwell's template schemata.
    - Schank's story skeletons seem to be of higher complexity.
    - Both Schank and Bordwell define similar processes on these structures (discussed later).
    - According to both Schank and Bordwell these similar structures originate from culture.
    - According to Schank communication is hardly possible without these structures.
- As an example of a template schema Bordwell specifies a master schema, consisting of the elements: *introduction*, *explanation of state of affairs*, *attempts*, *outcome*, and *resolution*.
- This schema is common in western culture; it is also very similar to Dautenhahn's preverbal transaction format.
- Events in the template are in chronological order and linear causality.
- Prototype schemata are similar to template schemata but represent characterizations of smaller structures like agents, actions, goals and locales.
- Using procedural schemata the fabula is dynamically constructed from template and prototype schemata.

4.5.3 Making and Manipulating Stories

Topic

- The process of fabula construction is further discussed here. Bordwell defines dynamic ('on-the-fly') story construction while Schank defines some higher processes that occur after the initial construction of the story.

In Schank's work, a story (the index) is made when a person what's to tell it. The events are already stored in some way and seem to be present all at once. He doesn't specify how these get into memory, if this is already a story or if it's some perceptual copy of the events or story told. Where Schank starts of his theory of story construction and manipulating, Bordwell's theory seems to end. Bordwell's defines the way the story constructed dynamically while it happens.

Bordwell's procedural schemata are a set of protocols to dynamically acquire and organize information. Procedural schemata assign story elements to prototype

schemata and classify them within the template schema, when a template is inadequate procedural schemata can be used to adjust expectations.

Bordwell defines three types of procedural schemata, each of them represents a different kind of motivation that can be used to assign or place story elements:

- **motivational**: relevance to story necessity
- **realistic**: how things work in the world
- **transtextual**: mostly expectations from genre

A last one is the artistic motivation, which means that something is there for its own sake. This artistic motivation is considered as irrelevant in the context of this thesis. Bordwell argues that in films motivational and transtextual motivations are used most, realistic motivations are used as reinforcement.

When using the three kinds of schemata (template, prototype, procedural), a memory is constructed of the fabula. Furthermore assumptions, inferences and hypotheses are made. These are all quite similar and have in common that they are subject to correction; also assumptions and inferences can be seen as hypotheses. Assumptions and inferences seem quite trivial and are easily and unconsciously made by humans. Assumptions are made about where someone is when this person is not mentioned in the story at that time, or that an individual has the same identity in subsequent appearances. When someone has to cry, the inference that this person is sad is easily made.

A more interesting and active process is that of hypothesis making. According to Bordwell constructing the fabula can be seen as a continuous process of anticipation about what will come next. Humans are tuned to monitoring unexpected change; continuity will fail to register after a while. This kind of hypothesis Bordwell calls a ‘suspense hypothesis’.

Another kind of hypothesis is the curiosity hypothesis, which is a hypothesis about something the syuzhet doesn’t specify. This type of hypothesis relates well to the importance of having (and generating) interests as identified in section 4.4.2. This will be left for further research (refer to chapter 8: Conclusions); the focus here is mainly on the suspense hypothesis, as Bordwell argues that this is the most important one.

A hypothesis can be of different gradations:

- **probability**: less or more.
- **exclusivity**: multiple simultaneous hypotheses possible or can replace each other successively, can be either/or set or a mixed set,
- **duration**[2]: long term (whole narrative) and short term (episode), assumptions and inferences are shortest term.

[2] Note that the property of duration is not explicitly listed with the properties of probability and exclusivity by Bordwell.

The initial portion of the story (syuzhet) establishes hypotheses that will be used to order subsequent information as long as possible. Very strong counter evidence is needed in order for the perceiver to withdraw these hypotheses. While Bordwell doesn't mention this, the early statement of the goal could give rise to these initial hypotheses. Also the selection of the initial template, master schema or skeleton could depend on these initial hypotheses, which would make withdrawing it indeed difficult.

In the end a hypothesis can be:

- **validated**: usually most hypotheses are validated
- **invalidated**
- **left dangling**

Hypothesis making isn't a clean and scientific process. The perceiver is eager to build upon his hypotheses and employs a wait-and-see strategy when looking for validation of a hypothesis. The latter means that pauses are seldom taken to figure out pitfalls in actions relating to a hypothesis, the perceiver rather rushes on. The question is if this is only valid in relation with films or not.

In Bordwell's theory the syuzhet is 'made' by the producer in order to play with the human need for anticipating on the story, this aspect of entertainment is not relevant in the context of this thesis as the syuzhet is not necessarily 'made' and stories are not used for entertainment.

Schank doesn't define a similar process like Bordwell's as specified above. His process of *distillation* seems to be the closest. Though, distillation only takes place when the story is to be told ('communicative intent') and he doesn't define the process in detail. Schank says that descriptions of physical items are left out, as well as particular words and ideas of the participants of the story. The index seems to be a generalization of the events, which draws upon a prior 'unstructured' memory of these events. Bordwell's theory could be used here to argue that the actual story construction already took place before telling it, at least to certain extent.

The question of whether or not Schank's definitions of story construction apply only to a generalized version (index) or the fabula will be left unanswered for a while. According to Schank indexes can be constructed in different ways. The basic one is distillation. Distillation arises from a communicative intent; the index is constructed deciding what parts to leave out and what parts to keep, this process of index construction if followed up by *translation*. This process translates the index to natural language and adjusts it to the environment of the teller. An index can be expressed in different intellectual levels and can be expressed to satisfy different goals, to show different points and to relate to different audiences

Furthermore Schank defines some manipulation processes that can be applied to prior formed indices. Combining two stories is the major process here, combining stories is combining two points of a story. This means leaving out events that don't make the point, called *suppression*. Then the two stories must be weaved into each other by first deciding which is the dominant story and then adding details from the other story that contribute to the point of the dominant story, this is called *conjunction*.

The previous *combination processes* are one example of *elaboration*. Depending on the environment and emotional impact needed, the teller can elaborate a story in different ways:

- **detail addition**: add details to index through search, reconstruction and adaptation in order to hold centre stage longer.
- **commentary**: add own commentary, not part of original index.
- **role-playing**: teller *combines* a story in order to add what a certain character might have said or felt.

Other processes Schank defines are:

- **captioning**: summarize larger story, if whole story isn't told the story will be forgotten.
- **adaptation**: take actors, plans, event and other elements from source story and add to analogous entities of target story.

Summary

- According to Schank a story is constructed when a person wants to tell it. He doesn't define how the story gets stored when the events are happening.
- Bordwell seems to define this dynamic process of story construction more clearly using procedural schemata and the process of hypotheses making.
- Procedural schemata dynamically assign story elements to prototype schemata and classify them within the template schemata, this way a fabula is constructed.
- There are three types of procedural schemata: *motivational, realistic and transtextual.*
- Humans are continuously anticipating what will happen next when constructing the fabula.
- There are *suspense* and *curiosity* hypotheses.
- Hypotheses can be of different *probability*, *exclusivity* and *duration*.
- Hypotheses can be *validated, invalidated* and *left-dangling*.
- Hypotheses made in de beginning of the syuzhet will be used to interpret subsequent information as long as possible, strong counter evidence is needed for the perceiver to withdraw these hypotheses.
- Schank calls the process that occurs when telling a story *distillation*.
- When a story is constructed, Schank defines some 'higher' processes like *combining* and *elaborating* stories.

4.5.4 Connecting Fabula and Index

Topic

- In addition to the similarities between skeletons / templates and Bordwell's and Schank's theories on story construction some additional differences and similarities will be pointed out between their theories.

Now Bordwell's fabula and Schank's index have been discussed their theories will be brought together and some problems will be identified.

Schank's index can be of different levels of generalization, but it is a generalization nonetheless. It not only has to function as a story, but at the same time as an index. In order to be able to retrieve 'similar' cases there must be some generalization or they will never be matched; it's highly probable otherwise that they will differ in at least in one event. The sections 'generality versus particularity', 'index as a viewpoint' and 'communication' will point out further differences and a way to still incorporate both views in one.

As can be seen in the previous chapters the fabula and index have certain things in common too, as they probably should have, because they are both representing the constructed story. If the structures were to be matched, the elements goal, plan and result should somehow map to the fabula. The other elements seem to convey other information. As can be seen both structures incorporate the notion of goal, in both their theories the goal is used as a starting point in the story. This conforms well to Bordwell's argument that the hypotheses made in the start of the story are the strongest and need very profound counter arguments in order for the perceiver to withdraw them.

The similarities between their concepts of templates, story skeletons and processes of manipulating stories were already pointed out in the previous chapters. The sections 'coherence' and 'meaning' will point out some more similarities between the fabula and the index.

Generality versus Particularity

The question here is how much of the actual events and their sequence do we really remember? Schank's index element 'plan' could contain a (short) sequence of events, though his examples don't suggest this. Also the word 'plan' suggests that this is an extracted/generalized form applicable to new situations.

The importance of the fabula and the index lies probably in different domains. Bordwell's fabula provides a standard way of interpreting the syuzhet, this already gives the events interpreted some meaning by putting in a certain (causal) order. Schank's generalization could give a more specific meaning to these events.

The fabula can be used to reconsider the lessons learned from the original events, because they are still stored in a non-generalized version. The index can be useful to apply to new situations as it is of a more abstract level. The index can be used as a starting point for adapting a story to a new environment, as was explained in the

previous chapter. Also the index can be used as particular viewpoint on the generalization.

Index as a Viewpoint

While in Schank's view the index is some general structure to which an emphasis must be given in order to convey a message, it seems plausible to argue that the more one generalizes the story, the more one seems to steer to a certain meaning. Leaving out things means deciding what's important and what's not.

Looking at indices this way, they could represent a certain viewpoint on the story. Because more indices are available for the same story according to Schank, it is possible to get reminded of the same story by different indices; the story can represent an illustration of several viewpoints.

Taking this even further to Bordwell's theory, these viewpoints can be seen as hypotheses. This way an index on a story can become invalid when its generalization turns out to be invalid. Also an index can represent an unanswered question concerning the story. This unanswered question can represent an 'interest' as mentioned in section 4.4.2, which guides the perceiver in finding new stories. As said several times before, more research is needed on this.

Communication

Schank argues that a gist is only constructed of certain events (temporarily stored in memory in an unstructured way) when there is a communicative intent. When a gist is constructed certain events are left out, and other are kept. This idea is in accordance with his argument that telling stories is what makes you remember and crystallize them. If this is to be extended to Bordwell's fabula this means that telling the story helps constructing the story, though Bordwell's view of constructing the story can be seen as constructing it by telling it to oneself.

Coherence

Schank mentions that forming an index, is finding a certain coherent whole from a memory of events. The same can be found in Bordwell's theory, finding a coherent whole is equivalent to finding meaning. The narrative information from the syuzhet is checked for consistency – whether or not certain events hang together, and for relevance – if an event is significant in the story as a whole. Different processes for manipulating stories play a role in finding coherence, which – as said – are discussed in section 4.5.3. Even if index and fabula are inherently different, this concept can be seen as finding coherence at different levels of abstraction. That is, finding coherence at the more abstract index, and finding coherence at the fabula level, which includes specific events.

Skeletons and templates assist in finding coherence because they form a basis of selecting certain events and leaving out others. The perceiver's persistency of anticipating is his struggle for finding coherence using this and other tools, which he doesn't always reach (dangling hypothesis).

Meaning

Schank's index / gist and Bordwell's fabula have things in common, as they probably should have, because they are both representing the constructed story. It is obvious that by generalizing a story in an index, a certain meaning or interpretation is given to these events. It represents a way in which the events are looked upon. Bordwell argues that the same process is going on when constructing the fabula. Events are, for example, put in a sequence that makes sense to the perceiver, which also colours the way these events are viewed. The index can then be viewed as even a more abstract, generalized form of this colouration.

Summary

- In both the index and the fabula the notion of a goal plays an important role.
- The strength of the fabula is that new meaning can be derived from the original events, because these are preserved.
- Because the index is a generalization of the original events it would match more easily to new situations, allowing for old stories to be applied to new situations.
- By connecting multiple of Schank's indexes to a story (Bordwell's fabula), each of these indexes can represent a certain viewpoint on the story.
- In the chapters Internal- and External Use 'coherence' already played an important role, making a coherent story (fabula) also plays a major role in Schank' and Bordwell's theories on the index, fabula and story construction.

4.5.5 Syuzhet

Topic

- Bordwell's syuzhet is explained.

Finally Bordwell's concept of the syuzhet is discussed up till certain extent. In Bordwell's theory the syuzhet is meant to entertain the perceiver, which is done by complicating the perceiver's construction of the fabula. In the context of this thesis, a multi-agent environment, the syuzhet could prove to be much more straightforward. Also the concept of 'space', which the syuzhet can cue, is of less importance to the application of this thesis. Some could be of importance in a multi-agent environment; these are briefly discussed here.

First of all, the definability of the syuzhet Bordwell suggests makes it interesting to have a better look at it. According to Bordwell the syuzhet is a system composing events according to specifiable principles. The syuzhet defines an 'extraverbal' logic, independent of any medium.

Another aspect possibly making the syuzhet more usable in this thesis is that Bordwell doesn't think that an implied author is essential for constructing the fabula

of the syuzhet. The narrative doesn't have full control over the perceiver's fabula construction though most perceivers will generally construct the same kind of story. The syuzhet merely gives cues, or signposts, to assist the fabula construction; an implied author is optional.

There are three principles that relate the syuzhet to the fabula:

- **narrative logic**: narrative defines what counts as an *event*, *cause*, *effect*, *similarity* or *difference*, this way the syuzhet facilitates in constructing causal relations in the fabula.
- **time**: narrative cues to construct fabula in any *order*, *duration* and *frequency*.
- **space**: syuzhet cues fabula space; not specified further in this thesis.

Furthermore the information of a syuzhet can vary in:

- **quantity**: is there enough to construct the fabula?
- **degree of importance**: what is relevant for the fabula and what not.
- **formal correspondences with fabula**: selection in the syuzhet creates gaps in the fabula; combination creates composition.

For each of the three principles that relate the syuzhet to the fabula (as mentioned before) the syuzhet can create gaps in this relation, resulting in causal, temporal and spatial gaps. Within these three types of gaps Bordwell gives some ways in which these gaps can be presented, through which different effects can be reached on the perceiver's tendency to make hypotheses. The different kinds of motivations stated before and represented by procedural schemata are used to fill in these gaps.

- **temporary / permanent**: temporal (filled in later) gaps build up surprise, permanent gaps make us scan back for missed information
- **diffuse / focused**: a focused gap (presented in a clear-cut way) solicits exclusive and clear hypotheses, diffused gaps give rise to open-ended hypotheses.
- **flaunt / suppress**: a flaunted (presented explicitly) gap shows possible importance of it later on, a suppressed gap is mostly used to mislead perceiver.

According to Bordwell these gaps are used to create retardation, delaying the revelation of information. The information can afterwards be given at once or in a more gradual way, this he calls exposition.

The question here is if the events perceived are not meant for entertainment, would these gaps still arise and have the same meaning? It could be argued that people naturally engage in filling in gaps, not only in storing telling. Its importance could be

beyond story perceiving, extending to perceiving real-world events that are not deliberately organized but still convey meaning in their incompleteness. A software agent trained in anticipating would handle incomplete information better and even derive meaning from it.

Finally, a syuzhet can emphasize certain parts of the story by giving redundant information while in other places information is scarce. The syuzhet can do this in several ways:

- **knowledge**: it is common to restrict knowledge communicated to what one character knows, but more overall knowledge beyond the character's can be added. Also knowledge can vary in subjective and objective depth.
- **self-consciousness**: narrative explicitly addresses perceiver; not particularly relevant in this thesis.
- **communicativeness**: story implies there is more knowledge to be communicated, but refrains from actually doing that.

Reality

Note that the syuzhet can be seen as a representation of Schank's objective reality. A separation is made between what's happening inside (fabula) and outside the mind (syuzhet). The property of communicativeness also suggests this; it gives the impression that there is information beyond the information in the story, even though Bordwell talks about fiction. This suggests there is some absolute truth, or in Bordwell's case, that the narrator knows 'the truth'. About the former can be said that everything there is and we need to know about is human reality, as argued before. The latter doesn't have to be true either as the narrator also may not know the information left out.

Summary

- In contrary to Bordwell's theory, entertainment is not an important aspect of the syuzhet in the case of a multi-agent environment.
- The syuzhet defines an 'extraverbal' logic, independent of any medium.
- An implied author is not essential for the syuzhet.
- Most perceivers will generally construct the same kind of story from the same syuzhet.
- The principles of *narrative logic*, *time* and *space* relate the syuzhet to the fabula.
- A syuzhet can vary in *quantity of information*, *degree of importance* and *formal correspondences* with the fabula.
- The syuzhet can influence the perceiver's hypotheses making process by leaving *gaps* in the information.
- The syuzhet can emphasize certain information by giving *redundant* information.

4.5.6 Overall perspective

Topic

- Some additional relations between the chapters External- and Internal Use and this chapter (What is the story?) are given to show that important properties in these chapters come back in the more concrete theory of this chapter.

Several important aspects of narrative in the previous chapters have explicitly come back in this discussion on narrative structure: individual coherence, reality, communication, manipulating stories, indexing and intelligence, language dependency, culture, intelligence and interests, generality/abstractness versus particularity.

Some other relations with pervious chapters that are not mentioned explicitly will be briefly pointed out here.

Culture & Coherence

How story construction involves finding coherence was already pointed out. A skeleton/template is chosen which already has a major influence around what point the story will cohere. The choice of this skeleton can be seen as the choice of how an individual wants to define his 'self', which represents individual coherency.

Group coherency can arise from the stories of ones culture; culture suggests several stories representing the coherency of the group. The same way, stories that go around a lot in a certain community will end up generalized and result in a skeleton. When a skeleton actually becomes culture and generally accepted is an open question.

Social Dimension & Communication: Dautenhahn

Much related to group coherency and culture is the social, communicative dimension of narrative. It is possibly that not only culture is the major construct in this. As pointed out in section 4.3.2 Dautenhahn suggests social origins of narrative structure [8: page 253]

- **introduction of characters**: making contact between individual, actors, listener and speaker.
- **develop plot**: is sequence of actions that convey meaning: value, pleasurable, unpleasurable.
- **high point and a resolution**: reinforcement or break-up of relationships.

It can be seen that there almost is a one-to-one mapping with Bordwell's master schema, to which Dautenhahn gives a social parallel inspired by preverbal grooming of primates. As already pointed out, this suggests that Bordwell's master schema is not only cultural common but common to humans. It's human's social nature that gives rise to the initial story skeleton/template. Further extensions and alterations of these can then be seen as cultural specific. Note that Schank gives examples of more specific skeletons, while Bordwell keeps his discussion to the simple master schema.

Skeleton / Template

Summarizing, the skeleton / template seems a concept of much influence in story construction. A story skeleton represents canonicity and genericness of a story, it assist in the coherence of the story and it is an important construct of culture. This way it plays both a role in finding individual coherence and group coherence.

Finally skeletons help interpreting new stories in terms of old ones, they convey basic knowledge of a story to interpret other ones. The generalized part of Schank's index, especially the observation, is an important ingredient for the construction of a skeleton.

Note that a skeleton doesn't always have to be used, though in Schank's view a story needs to be based on a skeleton in order to communicate it.

Sengers

Coming back to the narrative properties listed in section 4.3.1 a lot of analogues can be found with this chapter. Although Sengers interpretation is used in this thesis, the properties were initially defined by Bruner. Also Dautenhahn seems to use Bruner's work as a basis of her work, her preverbal narrative format is adopted from Bruner and she refers to the same properties as listed by Sengers.

First of all Sengers gives a notion of the balance between the tension of a story being 'normal' (canonicity) and the story behaving unexpected (breach) which conforms to the need for anticipation identified by Bordwell. The need for anticipation in Bordwell's theory is mainly played upon through the use of gaps, which in their turn induce hypothesis forming.

Sengers talks about three aspects that bring certain expectations to story interpretation. First there is *genericness*, expectations originating from culture. Furthermore Sengers talks about *normativeness*, certain conventions the story has to conform to. Both could be the result of story skeletons, the former more obviously. Bordwell also argues there is a more or less standard way of interpreting narratives, which could relate to the concept of normativeness. Expecting too much from a human's anticipating capacities could result in the human not being able to form a story.

Sengers also mentions *context sensitivity*: a story can only provide cues to narrative, a term that Bordwell also uses in his theory. Sengers argues that the reason that a narrative can only 'cue' is that a narrative is perceived with respect to the perceiver's lived experience.

Finally the property of *referentiality* is interesting to mention, which means that a narrative only has to stand up to its own subjective tests of realism. An agent doesn't need an objective world model; the narrative defines its own 'logic'. This could relate to the principle of 'narrative logic' of a syuzhet that is included in section 4.5.5, which seems to say the same in a more specific way. If Bordwell's syuzhet is extended to real-world events, it could be argued that there is an implicit logic in perceived events.

Summary

- The following important aspects of narrative identified in the chapters External- and Internal Use come back in the more concrete theory in this chapter: *individual coherence, reality, communication, manipulating stories, indexing and intelligence, language dependency, culture, intelligence and interests* and *generality/abstractness versus particularity*.
- It is explained how skeletons play a role in finding individual and group coherence.
- Dautenhahn gives a social oriented explanation of a narrative structure that is very similar to Bordwell's master schema.
- The similar concepts of skeletons and templates play a major role throughout the story (fabula) construction process.
- The narrative properties from Sengers listed in chapter External Use are related to the theory in this chapter.

5 Agent Architecture

Before continuing with the pit-game agent the underlying architecture will be given a closer look.

This thesis finds its roots in Software Agents Research by the Software Engineering and Collaborative Modelling Laboratory (SECML) of the Information Science Department of the university of Otago [35], with a focus on multi-agent interaction technology. This research provided most of the necessary implementation tools for this project.

At the start of research on this thesis a significant amount of work was done on a FIPA-compliant [12] multi-agent platform called Opal [28]. In addition to this a Java-based, object oriented Petri net simulation framework called JFern [23] was made.

Both have been used as a basis in several research projects on multi-agent technology [13, 26, 27] and will also be used as an implementation framework for this thesis.

The definition of Coloured Petri nets (CPN) will be given as it plays a main role in this implementation framework. JFern is explained in the context of Coloured Petri nets.

Coloured Petri nets will be, amongst others, used for controlling the behaviour of agents. This specialized Opal agent designed for this will be discussed subsequently.

Furthermore the transport service JXTA [19] will be discussed briefly. This transport service played an essential role in the initial concept of the pit-game in the context of multi-agent interaction technology research [26, 27] but will be of less importance for this thesis.

5.1 Opal Platform

Topic

- Short outline of the multi-agent platform Opal.

The (Java-based) Opal architecture [28] supports the use of agent-oriented concepts at multiple levels of abstraction. At the lowest level are micro-agents, simplified agents that can be used for conventional, system-level programming tasks. More sophisticated agents may be constructed by assembling micro-agents. The architecture therefore supports the systematic use of agent-based concepts throughout the software development process. The micro-agents can be used in the construction of more complex agents that are based on the Foundation for Intelligent Physical Agents (FIPA) specifications.

The pit game used in this thesis is a model of an open economic trading environment that is highly distributed and sometimes unreliable. Opal fits in here because a multi-agent system can provide a robust and scalable infrastructure in such an environment. Within an agent-architecture it’s also possible to replace agents by an improved agent to introduce improvements or to provide the agent with extended functionality.

Summary

- As multi-agent platform the java-based Opal architecture will be used.
- In Opal system-level micro-agents can be combined into a FIPA-compliant agent.
- An agent system provides a robust and scalable infrastructure.

5.2 Coloured Petri-nets

Topic

- A definition will be given of Coloured Petri nets. Furthermore a java-based Petri model will be given and it will be shown how this java-based model can be used within the Opal platform.

Petri nets were originally formulated in 1962 by C.A. Petri [25] as a theoretical mathematical construct. It is now recognised that Petri nets can be thought of as mathematical formalism for describing distributed, concurrent systems. Just as finite-state automata are an appropriate tool for describing sequential systems.

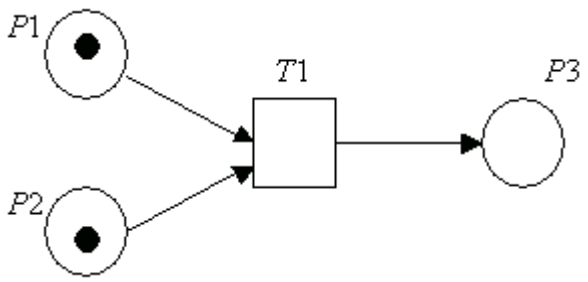
Although FIPA uses AUML [24] to represent its standard interaction protocols, Coloured Petri nets (CPN) are used because they are suitable for specifying concurrent conversations. By allowing multiple tokens in a CPN modelling an interaction protocol each token can represent one ongoing conversation. This way an interaction protocol defined in a CPN uses a minimal amount of system resources and is easier to handle.

The availability of analysis tools [11] allows for checking Petri nets for undesired loops and deadlock situations, which helps eliminate human errors introduced in the design process.

In the following chapter a precise definition of a Petri-net is given.

5.2.1 Definition

Petri nets are typically described in the form of a graphical representation, but the basic structure of a Petri net can be formally defined by a 5-tuple $(P, T, I, O, M0)$. An example of a graphical representation of a Petri net is:



- $P = \{p1, p2, p3\}$, a set of *places* (circles).
- $T = \{t1\}$, a set of *transitions* (rectangles).
- $I(t1) = \{p1, p2\}$, mapping of *input places* to a transition (arrows pointing to transition: *input arcs*[3])
- $O(t1) = \{p1, p2\}$, mapping of a transition to its *output places* (arrows pointing to places: *output arcs*[3]).
- $M0 = \{1,1,0\}$, the initial *marking* indicating the initial number of *tokens* in each place (tokens are in $p1$ and $p2$).

Note that the state of the system in a Petri net is represented by the positions of all the tokens in the net structure.

The three most important types of Petri nets are Condition-Event Petri nets, Place-Transition Petri nets and Coloured Petri nets. The most basic form is the Condition-Event Petri net; the two other types are generalisations of this Petri net providing better modelling behaviour and power. The Coloured Petri net is the most advanced of the three and is used in this project. The other two Petri nets will be explained first, because they provide an incremental way of describing the essential properties of Coloured Petri nets.

5.2.1.1 Condition-Event Petri nets

In a Condition-Event Petri (CE-net) net a place is not allowed to have more than one token in a place. When all *input places* of a transition contain a token, the transition is *enabled*. An enabled transition may or may not fire; a transition can become disabled again before it can fire. If a transition fires, a *token* will be added to the *output place* and the tokens in the transition's input places will be removed. In the example given above, which is a CE-net, the tokens from the places $p1$ and $p2$ will be removed and a token will be added to place $p3$.

5.2.1.2 Place-Transition Petri nets

Place-Transition Petri nets (PT-nets) are the most common form of Petri nets in research literature. A PT-net has some additional properties with respect to a CE-net:

- Places can have *more than 1 token*.
- *Arc's are weighted*, identifying the number of tokens consumed or produced along the arc when a transition fires. Default weight is 1.
- *Places have capacities*, representing the maximum number of tokens allowed in the place. Default capacity is infinite.

[3] In the formal definition of Petri nets there is no difference between arc's going to places and arc's coming from places. For convenience separate terms are used for them. Only in the java-based implementation of Petri nets in JFern an actual distinction is made between input arcs and output arcs.

These properties result in some additional rules for transition firing and enabling. A transition in a PT-net is only enabled when:

- for every *input place* of the transition the weight associated with the connecting arc is greater than the arc's weight.
- for every *output place* of the transition the sum of the tokens in that place and the tokens generated by the firing transition is equal or less than its maximum capacity.

When a transition fires, the amount of tokens removed from each input place equals the weight of its corresponding input arc. In the same way, the amount of tokens added to each output place equals the weight of its corresponding output arc. Generally the tokens removed from the input places are destroyed and new tokens are generated for the output places.

When the weights of the arc and the capacities in a PT-net are equal to 1, the net reduces to a CE-net. In general, when the places have a finite capacity, a PT-net can always be represented by a CE-net.

The power of a PT-net with respect to a CE-net is that it is better suitable for modelling concurrent distributed systems. In these systems often multiple independent interacting processes exist. Modelling these with a CE-net will result in a very large and hard to manage graphical representation. If there are two instances of the same process interacting in a CE-net this would have to be modelled using two separate and identical nets for each process. If a PT-net were to be used, only one net is needed and each instance of a process can be represented by a token, as multiple tokens are allowed in one place. By assuring that the places have a finite capacity it is still possible to use the analytical tools designed for CE-nets.

5.2.1.3 Coloured Petri nets

The Coloured Petri net (CP-net or CPN) as used in this thesis conforms to Jensen's formulation [18]. This CP-net is an elaboration of the PT-net as described above. The net structure of a CP-net is the same as that of a PT-net. The main difference is the introduction of data structures for tokens. In PT-nets tokens are of an unstructured single data type, in a CP-net this can be an abstract data type of arbitrary complexity. An abstract data type is called a *colour set*, a value of a token corresponding to a certain colour set, is called a *colour*.

A CP-net has a *global declarations* component specifying colour sets, constants, variables and functions which can be used on tokens of a specific colour set. Their scope is the entire CP-net.

As in PT-nets, arcs can consume or produce multiple tokens at once. In a CP-net a set of tokens consumed or produced is called a *multi-set*. A multi-set is a set that can convey repeated instances of an individual token.

The data (=colour) conveyed by a token can be used on arcs and transitions by specifying *net inscriptions*:

- Arcs have an *arc expression* that evaluates to a multi-set. In a way, arc expressions replace arc weights because the arc expression specifies how much tokens are consumed and produced by arcs. This will become clearer in the explanation of transition firing and enabling further on in this chapter.
- Transitions may have a *guard expression* that evaluates to a Boolean value. A transition can never be enabled if its guard expression doesn't evaluate to 'true'. The default guard expression evaluates to 'true'. Guard expressions may contain tokens.
- Transitions may have a block of computer code, called a *code segment*, which is executed when the transition fires. This code segment may use data from tokens but cannot alter this data.
- A place has an expression indicating its colour set. A place can only contain tokens of this colour set. This means that expressions associated with the output arcs belonging to one place must always evaluate to multi-sets of tokens of this colour set. Note that even though the arc goes *in* the place, it's called an output arc from the transition's perspective.
- Places may have an *initialisation expression* that can generate initial tokens in a place. An initialisation expression must evaluate to a multi-set of tokens conforming to the colour set of its place. The initial marking $M0$ (see CE-net) of a CP-net is determined by evaluating all the initialisation expressions.

The scope of an arc or guard expression is all arcs belonging to the same transition and the global declarations. Every variable (in a token or being a token) that appears more than once in this scope has to have the same value.

Another difference with PT-nets is that the places of a CP-net don't have capacities; their capacities are infinite.

Expressions, data types and functions on data types can be specified in any programming language. Jensen originally used the functional programming language Standard ML (SML) for this. In the context of this thesis a Java-based CP-net modelling tool called JFern is used in which the inscriptions are also specified in Java syntax.

The use of structured tokens and inscriptions results in a further complication of transition enabling and firing. When deciding if a transition is enabled, there are multiple ways in which tokens and variables can be assigned to expressions of guards and arc expressions. Each possibility is called a *binding*, more than one binding can result in the enabling of a transition. The following has to be the case when a binding is enabled:

- each input arc's expression must evaluate to a multi-set containing no more tokens than are present in its input place. If this is an empty multi-set, no tokens are needed in this place.
- the transition's guard must evaluate to 'true'.

There are no restrictions concerning the capacity of the output places, as there are none. When a transition fires, the following happens (in this order):

- the multi-set to which each input arc's expression evaluates is subtracted from its input place. If this is an empty multi-set, nothing is subtracted.
- the transition's code segment is executed.
- the multi-set to which each output arc's expression evaluates is added to its output place.

The benefit of a CP-net with respect to a PT-net can be illustrated best with the 'distributed system' example given in the PT-net definition. It gives an example of two concurrent interacting processes, with the two processes being identical. If the two processes now just have to be slightly different it is necessary to have two separate and slightly different nets for each process. This can introduce a considerable amount of redundancy. In a CP-net this can be solved by specifying additional information in the token about how it should be treated. The inscriptions of the CP-net can take this in account and behave differently for a certain kind of token.

The complexity of a CP-net can be distributed between its inscriptions, global declarations and its net structure. This allows one to model the same case in different ways. For example, a distributed system can be modelled with 1 input place, 1 transition and 1 output place. All the complexity of the model can be contained inside the arc inscriptions. When modelling the pit-game rules in a petri-net this resulted in some design decisions which are discussed in the introduction of section 6.2.2.

Again, a PT-net is a special case of a CP-net. Any CP-net can be converted into a PT-net, provided that it only uses colour sets with a finite number of possible values. This conversion, or *unfolding* can be quite complex.

Further properties of CE-, PT- or CP-nets are not specified here, as well as analyses algorithms, because they didn't apply directly to the research in this thesis. An example of a java-based CP-net is given in the next chapters where we introduce the CP-net design tool, JFern, used for the pit game.

5.2.2 JFern

In this chapter the most recent non-released version of JFern is discussed as this version is used for specifying the pit-game. The latest released version [22] only specifies the core objects which represent basic building blocks of the previously specified CP-nets in pure java. This version also had very limited viewing, editing and simulation capabilities. In the most recent version these capabilities are extended and reach their full potential. This version can be divided into three major parts:

- Core Objects: multi-set, place, transition, arcs, etc.
- Viewer: in the form of a 'Panel', 'Frame' or 'Internal Frame'.
- Editor / Simulator Toolbox: expands the viewer capabilities.

The core objects define the actual structural parts of a CP-Net in pure java. A net can be made by using these objects together in a Java source file. The viewer provides a way of loading a Java source file representing a net and viewing it in a Java program.

Finally the editor/simulator toolbox (JFern Toolbox) expands upon the viewer, adding simulating and editing capabilities.

Also a module was defined which provides a way of saving a JFern CP-net in a XML file. The module can convert a Java source file (compiled into an object) representing a CP-net to a XML file, and an XML CP-net file into Java source code. An additional XML file type was added to represent the layout of the CP-net in the JFern Editor. This layout file also accommodates additional annotation figures that can be used in the JFern Editor.

Some differences with the formal definition of CP-nets need to be pointed out. First of all, it is important to realise that the CP-net model was adapted to the java programming language instead of the other way around. Arbitrary java code is allowed everywhere in the java source of a CP-net. It is entirely up to the user of JFern whether or not to let the net conform to the CP-net specifications. However, when this is not done it is possible that certain properties and analyses tools for Petri nets don't apply anymore. The following things should particularly be taken into account in order to keep a JFern CP-net in conformation with its formal definition:

- In JFern only one general colour set may be used, which is of type Object. Tokens are always treated as being of type Object and places don't have a specified colour set (= object type). If the JFern user wants to use more colour sets he has to add type checking in the net inscriptions him self or assure by design of the net that only one type of object will ever enter a certain place.
- JFern doesn't provide explicit support for defining an initialisation expression for a Place. The initialisation code can be put somewhere in the net inscriptions or anywhere else in the Java source file.
- Tokens can be modified anywhere, but should only be generated in arc expressions. When defining a code segment (called *action* in JFern) it is important to ensure that the tokens are only read and not altered.
- Caution has to be taken when *Side-effects* are added to the JFern CP-net. Side-effects is code which is separate from the functioning of the CP-net. Side-effects should only be added to the *action* of a transition, as this is the only part of code that is assured to be run only once. If this code is added to an arc-expression it is possible that it is run multiple times, which could produce unwanted side-effects.

An explicit structural difference between the formal CP-net definition and a JFern CP-net is that a guard can be specified for an input arc. This feature is added because it doesn't alter fundamental properties of a CP-net and it enhanced the running speed of a JFern CP-net. Before checking if a transition is enabled, the input arc guards will be checked. This way it is not always necessary to generate all possible bindings for the transition, which can take up a considerable amount of time (sometimes by a factor 1000 or more). This is why input and output arcs are distinguished in the core objects of JFern.

JFern doesn't support the firing of a transition when there are no tokens. This is why a default guard inscription is added to every input arc, defining that at least one token is needed. This again prevents from numerous bindings having to be generated.

An additional benefit of the arc guard is that it can help making the net inscriptions more modular. Parts of a transition guard can be distributed to the arc guard if they only work on tokens generated by these arcs.

In the input arc expression, typically, tokens are selected from the input place. This can be done with the *var* command. A token can be named with *var("tokenname")* or tokens can be selected anonymously with *var(x)*, where *x* is the amount of tokens needed. If multiple named tokens are need this can be done with *var("tokenname1"); var("tokenname2")*, etc. There is no control over which token is assigned to which name. This is in conformation with the formal definition of a CP-net. The tokens selected with the *var* command will result in a multi-set accessible within the scope of the corresponding transition.

The scope of a transition includes all arcs belonging to the transition and the transition itself, in JFern the scope is called the co*ntext*. From this context a multi-set can be requested with the method *getMultiset()* or individuals tokens can be requested with the *get("tokenname")* or *getAny()* method. The *getAny()* method is used for requesting a random anonymous token.

An output arc expression can use these functions to get access to these tokens if necessary. The output arc expression should always return an object of type *Multiset* with the Java *return* command. This multiset can be generated with the java code

Multiset result = new Multiset();

Tokens (= objects) can be added to this multi-set by using the *Multiset.add(Object)* method. Additional code in the output arc expression can be as complicated as needed to generate the appropriate tokens.

In the next chapter it is illustrated how CP-nets generated by JFern can be used to control an Opal agent. This will be done using an illustrative example of a JFern CP-net.

5.2.3 Petri-net controlled Agents

A Petri-net specified in JFern can be used together with an Opal agent, for this a specialized Opal agent was defined, the CPN-Agent.

In order for the CPN-Agent to be able to use a Petri-net defined in the JFern package this Petri-net has to contain at least a placed labelled 'In' and a place labelled 'Out'. Through these two places a Petri-net can communicate with other Petri-nets run by a CPN-Agent. In the 'In' place the Petri-net can receive Java-objects and the Petri-net can place Java-objects in the 'Out' place to send them.

Optionally there can be a place labelled 'Start', when a CPN-Agent object (object-type 'CPNAgent') is created an initial token can be specified which will be placed in

this place. When a CPN-Agent object is created, the following parameters are required:

- A name for the CPN-Agent (for example “Klaas”).
- An xml-file defining a JFern Petri-net with a ‘In’ and ‘Out’ place.
- A type for the CPN-Agent (for example “Player”).
- Optionally an initial token.

When the CPN-Agent is activated (using the Opal-Agent method ‘activate’), the CPN-Agent uses the JFern package to read the xml-file and generate a JFern Petri-net. Next it will put the initial token in the ‘Start’ place, if present, after which the Petri-net will be activated. The Petri-net will now run like any other Petri-net defined in JFern, the only thing the CPN-Agent will now do is intercept objects that are being put the place called ‘Out’ and send them to other agents using the Opal transport service. Objects received by the CPN-Agent will be placed in the ‘In’ place of its Petri-net.

A design problem here was how the Petri-net can specify to the agents to which the object in the ‘Out’ place has to be sent. The solution requiring the least implementation time was chosen.

The Opal platform only accepts objects from the type ‘Message’, this object also contains a ‘receiver’ field in which the receiver of the message can be specified. The choice was made to use this Message object as the object type required in the ‘Out’ place and the type of the objects that will be put in the ‘In’ place. Messages placed in the ‘Out’ place should have their ‘ontology’ field set to the value ‘CPNToken’ in order for another CPN-Agent to place the in the ‘In’ place of their Petri-net.

This way code in the Petri-net has to perform the task of making a Message object from an internal object. Also when a certain object has to be sent to multiple receivers, the Petri-net has to generate a Message object for each receiver. Multiple receivers can mean a certain list, or a whole group of a certain ‘type’ (see 3rd parameter of CPN-Agent) in which case the Petri-net has to search for CPN-Agents of this type.

This functionality doesn’t really belong in the Petri-net because it is specific to the Opal platform. This is why it would have been better to modify the CPN-Agent to accommodate this functionality. The Petri-net should be able to only work with the objects of its ‘colour set’. Still some kind of interface needs to be defined so that the Petri-net can indicate if the object has to be sent to 1 agent, a list of agents or a group of agents conforming to a certain characteristic. This interface should be minimal and platform independent. Because of time restrictions this was not done within this project.

Summary

- Petri nets were originally formulated in 1962 by C.A. Petri.
- Coloured Petri nets are suitable for specifying concurrent conversations, using a minimal amount of system resources.
- Availability of analysis tools allows for checking Petri nets for undesired loops and deadlock situations.
- First a mathematical definition of Petri nets is given.
- In order of complexity Condition-Event (CE-), Place-Transition (PT-) and Coloured Petri nets (CP-net) are explained. Each type is an extension on, and can be converted to, the type explained before it.
- JFern is an adaptation of the CP-net model to the java programming language. It's up to the user to let the net conform to the original CP-net specifications, which is important in order to be able to use analysis tools.
- JFern consists of *Core Objects* representing the CP-net, a CP-net *Viewer* and an *Editor / Simulator Toolbox* for designing new CP-nets.
- A CP-net in JFern can be saved to a XML-file.
- The CPN-Agent is a specialized FIPA-compliant Opal agent on which JFern-based CP-net can be used.
- The CPN-Agent takes care of loading the JFern Petri net, initializing it and re-directing messages from and to (pre-defined) places within the JFern Petri net.

5.3 JXTA

Topic

- A short outline is given of the JXTA transport service.

JXTA technology [19] is a set of open protocols that allow any connected device on the network communicate and collaborate in a P2P manner. JXTA peers create a virtual network where any peer can interact with other peers and resources directly even when some of the peers are on different network transports.

The first of the two main services provided by JXTA is that of pipes, which represent an asynchronous and unidirectional message transfer mechanism used for service communication. The second service is that of advertisements. These are language-neutral metadata structures represented as XML documents and are used to describe and publish the existence of peer resources.

JXTA is used as one of the transport services available in Opal. The initial reason for implementing JXTA transport in Opal was to make Opal able to communicate with a wider range of devices such as PDA's and even cell phones. Later on when the Pit game was considered as a model for research JXTA's advertisement service became very appealing, as the FIPA standard didn't specify any kind of broadcasting service. The advertisement service could be used for announcing *bids*, which is a more elegant solution than implementing a broadcast service on a higher level.

The FIPA Agent Communication Language (FIPA ACL) doesn't have the notion of a group; the JXTA advertisement service can provide this. Currently when JXTA transport is used in Opal, one instance of the Opal platform will act as a JXTA peer and this peer will host all Opal agents run on the platform. Agents hosted by one peer form a group, although larger groups can be formed with other JXTA peers.

When sending point-to-point messages the appropriate transport service is chosen automatically by Opal. Messages going outside the Opal host can use JXTA (or HTTP); messages within the host can use for example Java-RMI. When an agent wants to advertise a message, JXTA will be used in both cases.

In the next chapter the context in which JXTA would be used will be clarified a bit more.

Summary

- JXTA is a set of open protocols that allow any connected device on the network communicate and collaborate in a P2P manner.
- JXTA has two services: pipes & advertisements.
- JXTA is available as a transport service on the Opal platform to make it able to communicate with a wider range of devices.
- JXTA's advertisement service is very suitable for announcing bids in the Pit game.

6 Pit-game Agent

6.1 The Pit-game

Topic

- The card game Pit is introduced.

The card game Pit [15] dates back to 1903 and is copyrighted by Parker Brothers Inc which is now owned by Hasbro. The game is based on the American Corn Exchange. The "pit" in question is "the commodities exchange". There are 63 cards representing

seven different commodities available on the exchange. Each commodity is pictured on 9 cards: barley, corn, flax, hay, oats, rye and wheat. The commodity cards show a picture of the exchange in action, a point value for the card, and the name of the commodity.

The Pit version depicted above is the Bull and Bear Edition. This edition contains two additional cards representing buyers. One card is the "bull" - the aggressive buyer. The second card is the "bear" – the conservative buyer. Owning these cards gives some additional restrictions and advantages.

In this thesis the basic (non bull and bear) version is used. In the next two chapters the rules of this version are explained and a motivation is given why this game is used as a test case.

6.1.1 Rules

Three to seven players may play and a dealer distributes nine cards to each player from a shuffled deck of cards. The deck is prepared so that the number of commodity types in the deck matches the number of players for the given hand.

When playing (one *hand*) begins (dealer strikes bell), the players independently and asynchronously exchange cards with each other, attempting to "corner" the market by getting all nine cards of any one type.

On any single exchange, they can only trade up to four cards from their own hands, and all the cards traded must belong to a single commodity type. Thus if a player has six barley cards, two wheat cards and one rice card, he will typically initially attempt to trade away his two wheat cards, hoping to acquire one or two barley cards.

Trading is carried out by a player (the "bidder") announcing, for example, that he has some cards to trade. If another player (a "trader") also wishes to trade the same number of cards he has to announce in return an equal number of cards, the two players then may make an exchange.

Whenever a player manages to get a 'corner', he announces that fact by striking the bell and calling out the commodity he got a corner on. After this the given "hand" is finished. Players who get a corner in 'wheat' (by getting all nine 'wheat' cards) get 100 points, a corner in 'corn' gets 75 points, in 'oats' gets 60 points, etc. The winner of the hand is the first player to collect 500 points.

6.1.2 Definitions

The rules give rise to some definitions that will be used in the rest of this thesis.

As said, in the card game players can announce asynchronously which cards they want to trade. The only rule is that both players must have announced the number of cards they want to trade. It is not defined who actually initiates the trading, that is, who announces first that he is willing to trade. It is assumed that the human players can work this out themselves.

In an agent environment this is not that obvious, and we have to make an explicit separation of roles (the first 2 definitions):

6.1.2.1 Bidder

The agent that announces a *bid* and possibly accepts an *offer*.

6.1.2.2 Trader

The agent that announces by means of an *offer* that he is willing to trade in reply to a certain *bid*.

6.1.2.3 Bid

The amount of cards a *bidder* announced he is willing to trade.

6.1.2.4 Offer

The message of the *trader* to the *bidder* that he is willing to trade the number of cards announced in the same bidder's *bid*.

6.1.2.5 Trade

The process started by the *bidder* sending cards after receiving an *offer* and ending by the *bidder* receiving cards sent by the *trader*. The latter in reply to the cards formerly sent by the *bidder*[4].

6.1.2.6 Hand

The session starting with the dealer dealing new cards and announcing the *hand* started and ending with the first *corner*.

6.1.2.7 Game

A succession of *hands* played. Points of each hand are accumulated throughout the game, the player with highest total points has won the game.

6.1.2.8 Corner

The first player to get all nine cards of any one type when playing a *hand* is said to have a *corner* and is the winner of one *hand*.

6.1.3 The pit game as an e-business simulation

First of all, at the University of Otago the Information Science department is interrelated with the Commerce department, in this context an e-business simulation fits well.

In order to treat an e-business example that covers the essential issues of interest but avoids extraneous matters, we look at a “closed-world” card game. The pit game provides a well-defined and simplified model of an open real-world trading environment allowing agents to interact autonomous and asynchronous.

Summary

- The rules of the card game Pit are explained informally.
- Translating these rules to a computer environment makes some definitions necessary: *bidder, trader, bid, offer, trade, hand, game* and *corner*.
- The pit-game is used as a simplified model of an open real-world trading environment.

6.2 Agent Design

As already pointed out in chapter 2.4, and further discussed in the next chapter (6.2.1, Original Design), the pit-game agent(s) available at the start of this thesis were not suitable for this task. In order to provide a suitable test-bed for this thesis new pit-game agents were designed, this new design is specified in chapter 6.2.2: New Design: Petri-Nets.

[4] Note that a bidder sending cards in reply to an offer can be seen as *accepting* the offer. This will become clearer in section 6.2.2.3.

Implementation details and possible future work are left out because they are not particularly relevant to the subject of this thesis. The implementations details are not needed to understand the design of the Petri-net.

There are still some things to be improved on especially the design of the player Petri-net (see section 6.2.2.3). The communication protocol of the agents should be formally worked out and carried through to the player and dealer Petri-nets, when the agents communicate in a fast pace some (synchronization-) flaws appear in the current protocol. Also some security issues should be resolved to prevent agents to be able to cheat on the game rules. Furthermore the current interface between the player Petri-net and the strategy (and with that the Petri-net itself too) should be extended so that the strategy has exactly the same options (choices) as a pit-game player in the real world would have.

These problems were communicated internally within the [35] and will not be included in this thesis.

As said in the introduction, in order to account for the substantial amount of time spent in this project on analysing the old design and making a new design this chapter (Agent Design) was included.

6.2.1 Original Design

Topic

- The implementation and shortcomings of the Pit game agents as they were present before this project are discussed.

At the start of this project a micro-agent implementation in Java was available of a more free interpretation of the pit-game rules. The implementation can be characterized through the following properties:

- Communication by function calls
- Hard-coded rules: no explicit definition of rules
- Turn-based: agents play synchronous
- Micro-Agents: not FIPA compliant
- Dirty Cards: no new cards were dealt for new round

Though the code in the micro-agents runs independent from each other, the agents communicated by function calls. The rules of the pit-game were not explicitly represented in the implementation, but were implicitly contained in the Java code.

In order to provide a base for a pit-game playing agent using narrative technology, the rules of the game should be defined explicitly and separately from other code by using Petri-nets. Specifying rules in this manner also permits new or updated rules to be sent to other agents, which fits in the context of other work in Software Agents Research done at [35]. This is not mentioned in here, as it goes beyond the context of this thesis.

Furthermore in the micro-agent implementation the agents communicated synchronously; they were given turns. In essence the pit-game is a game where people trade cards asynchronous. This also affected how autonomous the pit-game playing agents were, they could decide ‘what’ to do, but not ‘when’.

Opal agents provide a means of communicating with any other FIPA-agent. Also a basic implementation was available of Petri-Net driven Opal agents (CPNAgent, see section 5.2.3). While micro agents could be run asynchronous, they can’t communicate with FIPA-agents and no Petri-Net scheme exists for them. For these reasons, and that a Petri-net implementation is needed, it was decided to re-implement pit-game playing agents as Opal agents.

The Java code of the micro-agent implementation was not entirely made redundant by these decisions. In addition to some small parts of code, the following elements could be re-used in the Opal/Petri-Net implementation:

- Basic objects: Bid, Card, CardDeck, Cards, Commodity, Hand and PlayerID.
- Strategy Interface: Parts of it were re-used.
- Game and Player objects: Parts used, Game becomes Dealer in new implementation.
- GUI’s: Player and Dealer (Game)

Note that *Game*, *Player* and basic objects originated from a pit-game ontology on which the micro-agent implementation was based. In this ontology also some interaction protocols were specified. This ontology is not used for the new design of the pit-game agent(s), because it was an initial version and changed significantly in the micro-agent implementation. Also in the new design even more changes were made which made the initial ontology redundant. The objects mentioned above can be seen as a part of the ontology being adopted for the new design.

The basic objects could be re-used as they represent some basic elements of the game. These objects were slightly adjusted in the new implementation and more complex objects were built from them. This was especially due to the fact that instead of passing objects as arguments in function calls they had to be conveyed in a transport service provided by the Opal platform. An example of this is that ID fields in objects were often pointers to specific objects. When sending an object to another player the references object is not available anymore, an ID in the form of a string, integer etc. had to be introduced for these fields/objects.

The strategy interface allows new strategies to be connected to a pit-game agent. This interface was implemented with a java interface / implementation construct: a Java interface defines functions and an object which implements this interface should define implementations for all these functions.

Though in the micro-agent implementation the rules were not defined explicitly they were separated from the agent’s decision making process through the use of this strategy interface. When a micro-agent is confronted with several options during the game, it can call a function defined in the strategy interface. The implementation that is coupled to this interface at that moment will make a decision.

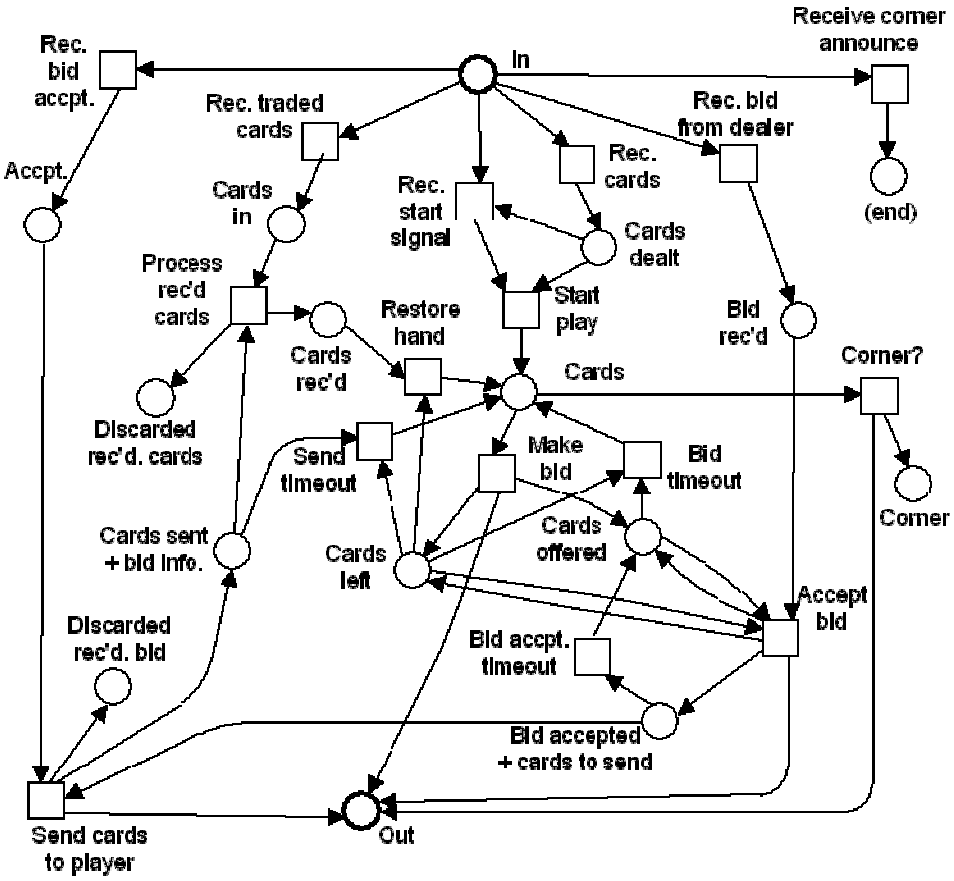
The strategy interface allows for different strategies to be connected to the same rules. This concept needed to be adopted in the new implementation as this provides a way of connecting a narrative inspired strategy to the pit-game agent. Also, because different strategies can be connected to different pit-game agents, experiments can be done about the performance of these different strategies. Note that when defining the rules in a Petri-net, and defining the strategy in a separate entity as well, both are exchangeable (see section 7.4).

The objects Game and Player represent hard-coded versions of the dealer and player. Though the pit-game rules are fully implemented in Petri-nets in the new implementation, some supporting functions and variables were useful in the new implementation as well. Examples of this are functions to register players, functions to retrieve player ID's, and score variables. The Game object will become the dealer Petri-net in the new implementation. The name 'Game' was suitable in the micro-agent implementation instead of 'Dealer' because it didn't deal cards and represented the game, passing every message and giving turns to each player.

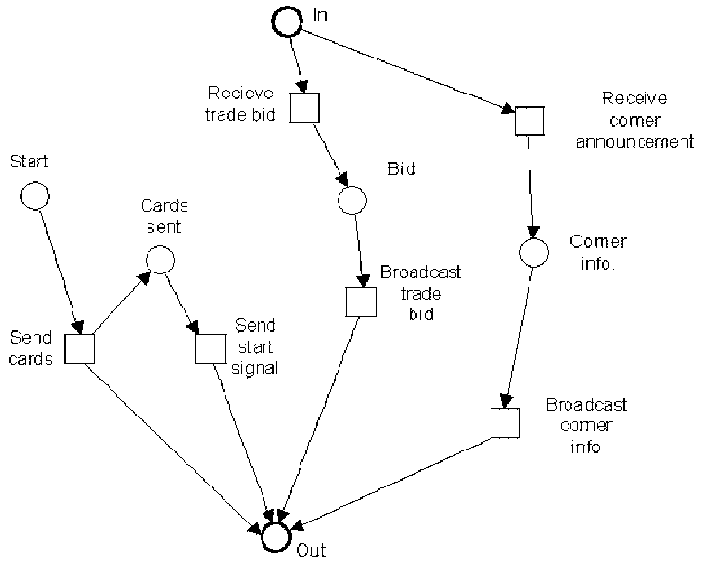
When a player had a corner in the micro-agent implementation the cards of his corner became dirty and couldn't count for another corner; the player first had to trade this cards away. This way the game could continue without reshuffling and re-dealing the cards after each corner.

Opal provides a GUI to show traffic between agents (amongst others) and the CPNAgent (see section 5.2.3) provides a GUI to show the Petri-net in action. The original micro-agent GUI's were carried over to the Opal implementation, because they provided some extra information about game status (scores) and the status of individual players (cards in hand).

In addition to re-used code from the micro-agent implementation there was an initial graphical representation of the pit-game rules in a Petri-net:



Also an initial Petri-net specification of the dealer was available:



As will be seen in the following chapters the graphical layout of the Petri-nets needed to be extended and changed significantly. Also no inscriptions were specified for both Petri-nets, which is the most time consuming aspect of the design of a Petri-net. In addition to solving problems in the design of the inscriptions of the Petri-net, the debugging of the inscriptions was complicated. The code is distributed throughout the net and had to be tracked down in the representing JFern/Java source file (see section 5.2.2).

Summary

- The Pit game agents that were present before the start of this project can be characterized as follows: *communication by function calls, hard-coded rules, turn-based, micro-agent based.*
- The new design should be: *Petri-net based* (JFern) and *implemented using Opal (FIPA-compliant) agents* to make them truly autonomous and able to communicate with agents on other (types of) platforms.
- Some basic objects of the old implementation could be re-used from the old implementation.
- There was an initial version of the player and dealer Petri-nets. When these Petri nets were implemented it turned out they needed significant modification and extension. Also no inscriptions were specified for both nets, this generally contributes to most of the work spent on implementing a (JFern based) Petri net.

6.2.2 New Design: Petri-Nets

Topics

- The new design of the Pit game agents is explained including important design decisions that needed to be made.

In the previous chapter already some constructs that will be used for the new design were identified. Before the Petri-nets and other specific elements can be designed some other decisions have to be made:

- Broadcasting of messages
- Offers & Trades Private or Public
- Management
  - Registering of players
  - Dealing cards

Both registration and broadcasting can be handled by JXTA. Registration can be realized through JXTA's concept of groups and broadcasting can be done through its advertisement service.

Dealing cards can be avoided through the use of 'dirty-cards' as used by the micro-agent implementation characterized in the previous chapter. If this concept is not used, new cards need to be dealt after every *hand* played.

A dealer can be used for handling broadcasting messages, registering players and dealing cards, this would be a centralized design; each pit-game player can also perform these three tasks locally. In this case registering player would involve a more complex protocol and the best solution for dealing cards would then be not to deal them, but to use the dirty-cards method.

The choice was made for the centralized design, using a dealer agent that broadcasts messages, registers players and deals/shuffles cards. JXTA was not used in this version though it was decided to program the dealer in a way that allowed substituting registering and especially broadcasting by JXTA in a straightforward manner. The dealer will be able to manage groups on a single Opal platform and will automatically register players marked as being a pit-game player agent. JXTA would extend communication and registering to agents on other computers anywhere on the world. This functionality was not required for the purposes of this thesis and would unnecessarily lengthen implementation time.

Also it was decided that one dealer represents one *game*. In principle more dealers can be instantiated to allow for more games going on at once. The registering system will have to be more complex for this. The player agent that will be designed for purposes of this thesis will not be able to sign up to a specific game, the dealer will automatically register players in its scope (the agent platform it runs on) if they are marked as pit-game player agents. The game for a newly arrived player agent starts at when the next *hand* starts. If a player agent doesn't have the right marking it cannot play.

Summarizing, two types of agent will be designed:

- **Dealer Agent**
  - manages 1 *game*.
  - automatic registering new players on platform.
  - broadcasting bids.
  - broadcasting corner announcements.
  - shuffling and dealing new cards starting a new *hand*.
- **Player Agent**
  - requires 1 dealer to be present on same platform.
  - bids and corner announcements sent to dealer.
  - offers and trades sent directly to other players.

In the remainder of this chapter the pit-game agents player and dealer will be called *player* and *dealer* respectively; both refer to computer agents.

Finally a decision needed to be made whether or not to make *offers* and *trades* public. In the micro-agent offers and trades are public. Because offers and trades are in principle between two players, this was not necessary. Though, in comparison with the original card game players can draw certain conclusions when other players trade with each other. A player can try to make assumptions about which cards were traded to whom for example.

The main part of this thesis is to define a new strategy based on narrative theory. Though this strategy could make use of the additional knowledge communicated through public offers and trades this was not considered as essential and will be left for future work.

All basic functionality needed for the *dealer* and *player* is modelled in a separate Petri-net for each agent. The only functionality left outside the Petri-net is the decision making, this is left to the strategy which is discussed in section 6.2.2.3 / basic

strategy. The strategy interface from the micro-agent implementation was changed considerably and is discussed in section 6.2.2.3 / strategy interface. The Petri-net restricts possible decisions to the rules of the pit-game. Values returned through the strategy interface are checked for whether or not they conform to the rules.

In the following chapters the functionality of the *dealer* and *player* will be explained with their Petri-nets made in JFern as a guideline. Not all functionality of a Coloured Petri-net is represented graphically; inscriptions (arc & guard expressions, etc.) also control the behaviour of the Petri-net. Because it is possible to represent any functionality from inscriptions also graphically the effort was made to find a balance between both. A major guideline was to represent the behaviour characteristic to the rules of the pit-game graphically, while more obvious behaviour was implemented through inscriptions. A brief explanation of this behaviour is given for each Petri-net. Furthermore important design decision are discussed, some arising from changing the initial Petri-net (previous chapter) to the new Petri-net.

Note that all the functionality of the pit-game agents is specified in the Petri-nets, they can directly function as an agent using the Opal-based CPN Agent as defined in section 5.2.3. In the case of the *player* agent some additional functionality is defined in a strategy object (section 6.2.2.3 / basic strategy), the strategy will be loaded automatically by the player agent when it is activated.

Before discussing the design of the *dealer* and the *player* a definition is given of the messages that can be exchanged between both agents.

6.2.2.1 Messages

The messages specified in this chapter are not Opal messages but the objects conveyed in the content field of Opal messages. Only one type of Opal message will be used, the content (the objects conveyed) defines the type of the message.

For the messages in the new design four basic objects were used from the old design: *Bid, Cards, Hand* and *PlayerID*.

Bid	**Cards**	**Hand**: *subclass of Cards*	**PlayerID**
bidID: *String* bidder: *PlayerID* numberOfCards: *int*	deckID: *long* cards: *List*	**inherited:** deckID: *long* cards: *List* player: *PlayerID* fullSize: *int*	name: *String* address: *String* info: *String*

Bid represents a *bid* from a particular *player*, Cards represents a set of cards separated from a Hand. Hand is a set of cards belonging to a certain player; it doesn't always have to be a full hand during the trading of cards. This is the reason the full size of the hand is stored in fullSize. The PlayerID object represents a particular *player* in a *game*; its fields convey information required to communicate with this player.

The object Hand is not to be confused with a played *hand* as specified in section 6.1.2.6. The similar naming can be justified as a *hand* starts with the dealing of a Hand and ends with the returning of the Hand when one of the players reaches a

corner. To the original Hand of the micro-agent implementation the fields ‘player’ and ‘fullSize’ were added; a hand belongs to a particular player and has a maximum size.

The field deckID in Cards and Hand is used by the player and dealer to check whether or not the cards received and sent belong to the current *hand* played. The deckID is changed for every new hand.

The four objects specified above were used to specify the message that needed to be exchanged for several purposes. For each of these purposes the particular combination of these four objects is given. Often the same combination of objects is used for different purposes. The value of a particular field determines what the object means. The value of this ‘key-field’ is included in the third column:

Starting & Ending Game	**Objects**	**Key-field**
receive game start announce corner **receive** game end	Hand Hand Hand	player = *this agent* player = *this agent* **won**: player = *this agent* **lost**: player = *other agent*
Bidding & Offering		
make bid make offer reject offer	Bid (Bid, PlayerID) (Bid, PlayerID)	bidder = *this agent* PlayerID = *this agent* PlayerID = *other agent**
Trading		
accept offer / send bid cards send offered cards	(Bid,PlayerID,Cards) (Bid,PlayerID,Cards)	PlayerID = *other agent* PlayerID = *this agent*

The composite object (Bid, PlayerID) is called **Offer**; the composite object (Bid, PlayerID, Cards) is called **TradeCards**.

Note that the values *this agent* and *other agent* are relative to whether or not the agent receives or sends the messages. Except for game start and end the values are given from the perspective of sending the object, values invert (this = other, other = this) when the object is received.

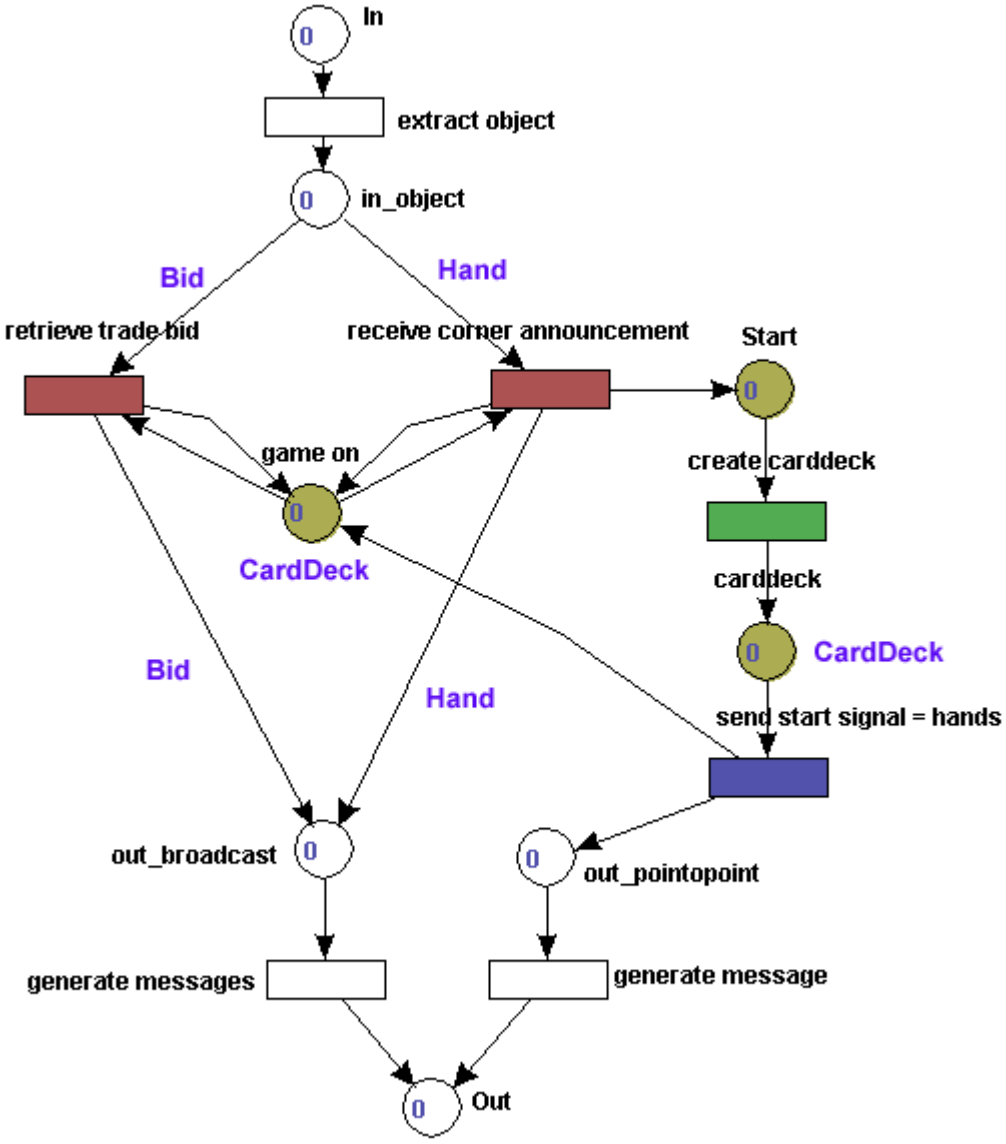
A received Hand object with its key-field set to the agent itself has ambiguous meaning; here the state of the pit-game determines what the message means. If the *player* is waiting for a new game to begin, it is interpreted as a game-start signal. If the player just announced it has a corner, the player interprets it as having won the game.

The so-called key-fields could be represented in the sender and receiver fields of an Opal Message object. Because the Message object is implementation specific and the sender and receiver fields were needed throughout the Petri-net it was chosen to represent them in the game objects (Bid, PlayerID). As will be seen in the *player* Petri-net a special transition connected to the ‘Out’ place uses these fields to construct the Message object and send it to the right Opal agent (a player or dealer).

The reject/accept messages already reveal a part of the protocol used for bidding, offering and trading. The *bidder* makes the decision if the trade is going to be made; this implies that when a *trader* makes an *offer* he is not allowed to withdraw it until an acceptation (the cards) or a rejection was received. The *bidder* can withdraw its bid anytime; it only needs to reject offers made on this bid after the withdrawal. This has implications on timing-out a *bid* or *offer*; this matter is further discussed in section 6.2.2.3. Note that rejecting an offer means bouncing back the Offer object formerly received without changing any fields.

6.2.2.2 Dealer

The dealer (coloured) Petri-net was (re-) designed using JFern as follows:



To the graphical representation generated by JFern the object types (colorsets) of the places where added in purple/bold. The non/white-coloured places and transitions are used to separate implementation specific functionality (relating to Opal) from the basic dealer-functionality. The ‘in_object’ place is the place where objects from other (CPN-) agents are received, objects that are placed in the ‘out_’ places will be sent to

other agents. Objects placed in the '_broadcast' place will be sent to every pit-game player on the platform (see introduction of section 6.2.2); objects placed in the '_pointtopoint' place will be sent to a specific player. How this is exactly implemented will not be further discussed in this thesis. This construction relates to some design decisions made in the design of the CPN-Agent, for more about this see section 5.2.3.

Because the white places (except 'In' and 'Out') can contain multiple types of objects, the object types in the figure are specified with the out- and input arcs belonging to these places. Note that this kind of usage of the places does not conform to the Coloured Petri-net specifications; one place can only contain one object type. To speed up the design of the Petri-net it was decided to nevertheless design the Petri-net like this, possible solutions are left for further research.

Places without an object type contain an arbitrary (dummy) object-type. Furthermore, red places receive data; blue places send data and green places do neither of both.

The labels of the places and transitions, and the added object types explain most functionality of the dealer Petri-net. Some functionality of the Petri-net, which doesn't show in the graphical design, needs some extra explanation.

The dealer Petri-net needs to be initialised with a dummy-token in the 'start' place. The dealer will create a CardDeck object, representing a new deck of cards. When creating the CardDeck object the number of players has to be supplied, the cards in the deck are generated in the same way as the original card-game does:

- Up to seven players: ***#suits = #players, #cards_per_suit = 9***

An additional formula was specified so the number of players could be increased. Because it was not perfected and for the testing purposes of this thesis no more than seven players were used, this formula is not specified here.

Each suit is of a certain type of commodity and each commodity has a particular value, cornering a high value commodity results in a higher score. If there are n suits ($s_i..s_n$), the value of a suit is calculated by $s_i = i * 10$. Note that the commodity values of the original game were 40, 50, 60, 70, 75, 85, and 100. These values give rise to some extra tactics; changing corner from one suit to the highest suit pays of more than changing to another suit. Because the game already gives rise to a sufficient amount of tactics it was chosen to increase the values in a linear way.

Because the composition of the deck cannot be changed while playing a *hand*, players cannot enter the *game* during a hand; they will automatically be detected when a new hand is started. The new deck created for the new hand will be composed according to the new number of players.

When the CardDeck object is created the cards are automatically shuffled, and the object can be used for generating new hands for each player. For each hand cards are separated from the deck and put in a Hand object, this object is put in 'out_pointtopoint' and will be sent to the designated player.

The remaining, empty, CardDeck object is places in the ‘game_on’ place to enable the dealer to broadcast bids and corner announcements. The CardDeck is used every time a Hand object comes in from the ‘in_object’ place in order to verify if the cards in this Hand object originated from this CardDeck. Bids are passed through to the ‘out_broadcast’ place immediately after receiving; a received Hand object will be checked if it indeed represents a corner. If the latter is the case the Hand object is broadcasted (‘out_broadcast’), the CardDeck is taken from the ‘game_on’ place and a dummy token is placed in ‘start’ place, indicating that a new game can start.

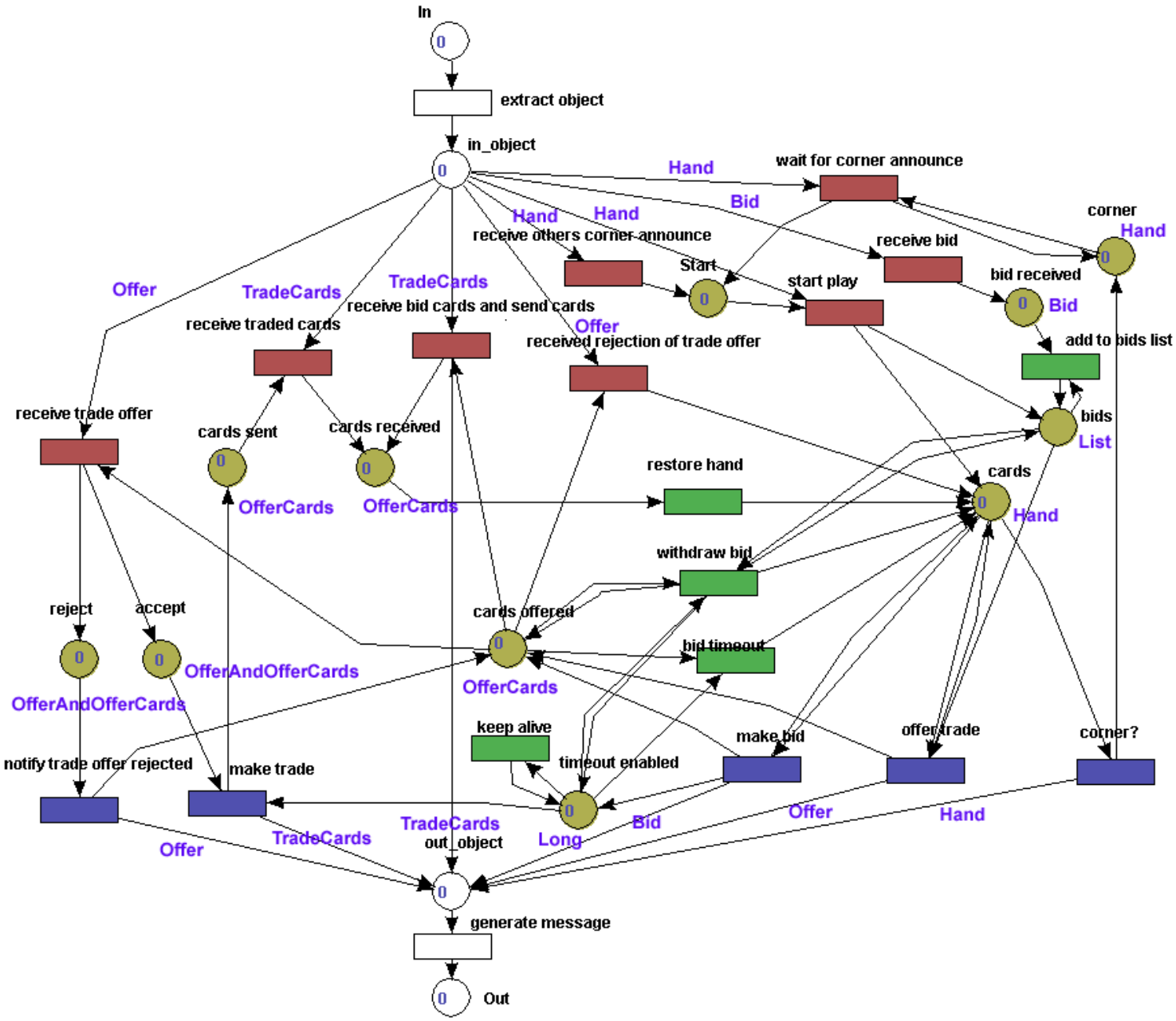
The dealer Petri-net also keeps track of the scores of the players; this is not modelled in the Petri-net itself and will not be further discussed in this thesis. The score of a *player* who reached a corner will be updated as follows:

- ***new_score = old_score + cornered_commodity_value***

In the original card game the player who was the first to have 500 points won the *game*. In the agent version the game in principle never ends.

6.2.2.3 Player

The player (coloured) Petri-net was (re-) designed using JFern as follows:



For the meaning of the colours in the Petri-net please refer to the description given with the dealer Petri-net in the previous chapter. First a description of the basic functioning of the *player* Petri-net is given, after which some important design decisions are discussed. In the description of the player Petri-net the strategy interface and its methods are referred to quite regularly, this interface will be discussed next. After that a basic strategy suitable for our testing purposes (see section 7.5.2) will be specified.

The player Petri-net introduces a few new objects that are only used within the Petri-net. The objects ‘OfferCards’ and ‘OfferAndOfferCards’ are composite objects; they consist of objects already defined in section 6.2.2.1. They needed to be defined in order to comply with the Petri-net syntax that only one colourset (data type) is allowed in a place. OfferCards consists of the objects Hand, Bid and Cards. OfferAndOfferCards in its turn consists of the objects OfferCards and Offer.

The OfferCards object represents an outstanding *bid* or *offer* within the player Petri-net. When the bidder field of its Bid object contains the playerID of the agent itself the OfferCards object represents a bid, when it contains the playerID of another agent it represents an offer (the offer is made on this bid).

Furthermore there two standard Java object types are used. In the place ‘timeout enabled’ a Long object is used to carry a 64-bit time-value, in the place ‘bids’ a List object is used to carry a set of Bid objects.

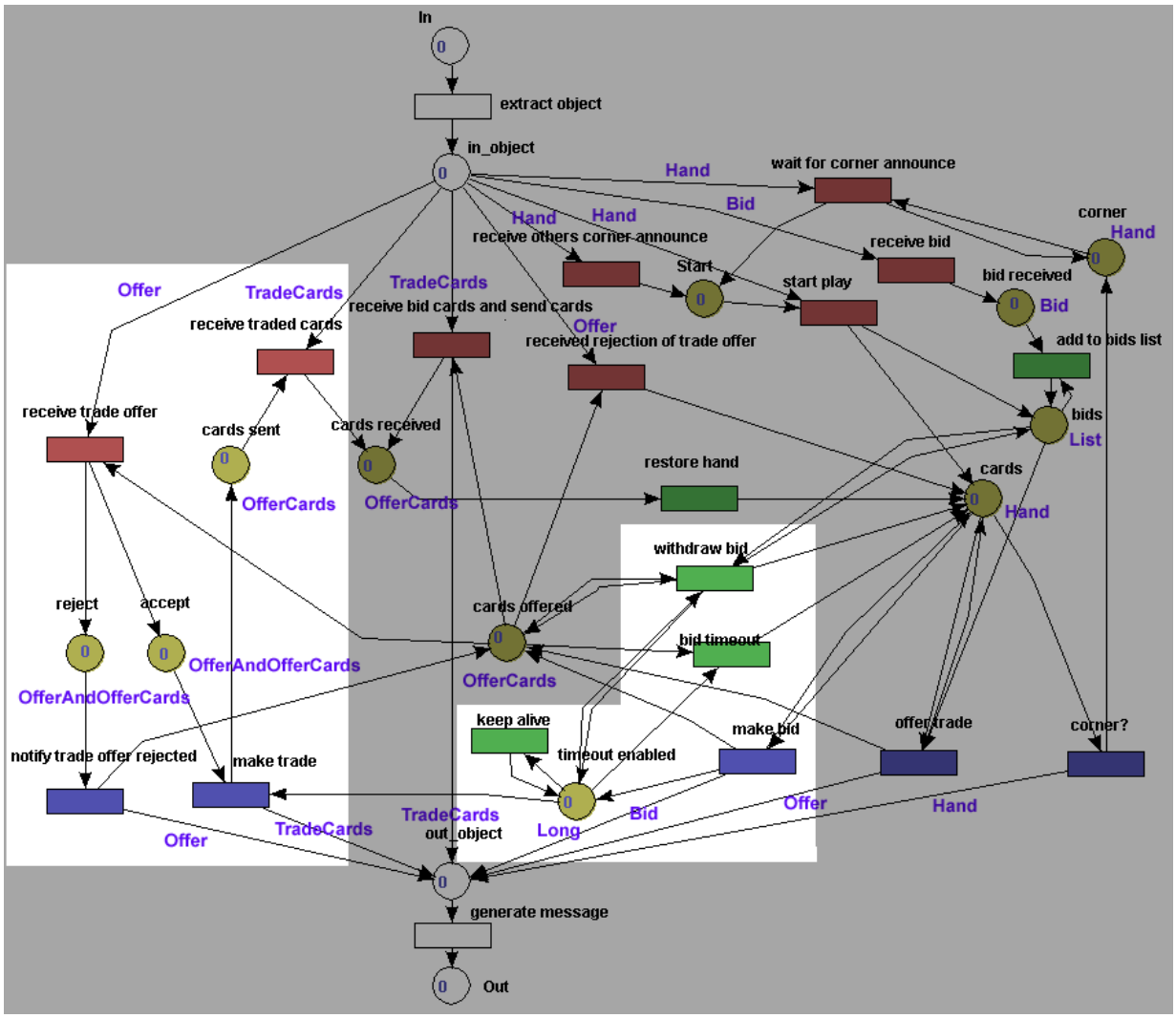
The player Petri-net needs to be initialised with a dummy-token in the place ‘Start’ place. A new *hand* for this player then starts when it receives a Hand object (see section 6.2.2.1) from the ‘in_object’ place. Transition ‘start play’ will put this Hand object in the place ‘cards’.

The player Petri-net is now enabled to make a *bid* with the ‘make bid’ transition, if there are *bids* in the ‘bids’ place the player Petri-net can also choose to make an offer instead. Note that there is no rule about which transition will fire first because Petri-nets are non-deterministic. This results in a problem because the strategy (see section ‘strategy interface’) will either first been given the option either to bid or to offer, it cannot make a decision between both when both are possible. To resolve this problem the player Petri-net and strategy interface need to be changed, this will be left for future research.

When the ‘make bid’ transition fires, the player Petri-net enters the *bidder* role. When the ‘offer trade’ transition fires the player Petri-net enters the *trader* role.

Bidder Role

First the bidder role will be discussed, this role is performed by the following highlighted places and transitions of the Petri-net:



The guard expression of the ‘make bid’ transition contains a query to the strategy interface whether or not a bid needs to be made and how much cards should be bid (getCards, see section 6.2.2.3 / strategy interface); when a bid should be made, a Bid object will be placed in the ‘out_object’ place. Then an OfferCards object is placed in the ‘cards offered’ place. This object contains the Hand object that is removed from the ‘cards place’, the ‘Bid’ object that was put in the ‘out_object’ place and the Cards object containing the cards that was return by the strategy interface. The cards in the Cards object are assumed to be removed from the Hand object by the strategy connected to the strategy interface; this is double checked by the ‘make bid’ transition.

Furthermore a time-out value is put in the ‘timeout enabled’ place, this time-out value represents the time the Bid object was placed in the ‘out_object’ place. The guard expression of the ‘bid timeout’ transition will pass this time-out value to the strategy interface (timeoutBid, see section 6.2.2.3 / strategy interface). By comparing the current time with the time-out value the strategy interface can determine the time the Bid has been out. The strategy interface can return a Boolean value to communicate if the Bid has been standing out long enough or not. When this is the case, the cards in the Cards object will be added to the Hand object. This Hand object will be placed in the ‘cards’ place. The Petri-net can again choose between the *bidder* role and the *trader* role.

Note that the transition ‘keep alive’ was added as a work-around because JFern only checks the guard of transition ‘bid timeout’ once. If the strategy interface now decided not to time the bid out, this guard will not be checked again. In order to make JFern check the time-out value continuously, the ‘keep alive’ transition keeps removing and putting back the Long object. This is its only purpose; it does not change the timeout value.

Now the OfferCards token in the ‘cards offered’ place represents the fact that there is an outstanding *bid*. In addition to timing out this bid, three other functions can now be performed by the Petri-net.

First of all, the Petri-net can decide to withdraw the bid, for this there need to be bids of other players in the ‘bids’ place. The decision to enable the Petri-net to withdraw a bid in reaction to bids made by other players is made by the strategy interface. In addition to making an *offer* on one of these bids, the strategy interface can also decide to make a counter bid, increasing or decreasing the number of cards in response to the other bids.

Withdrawing a bid is performed by the ‘withdraw bid’ transition. This transition queries the strategy interface (withdrawBid, see section 6.2.2.3 / strategy interface) to determine if the bid needs to be withdrawn. If the current bid (OfferCards in ‘cards offered place) is not withdraw the bids from the ‘bids’ place are discarded. Because the strategy interface didn’t withdraw because of them it is assumed that it is not interested in bidding on them. How the bids in the ‘bids’ place are maintained and added will become clear in the discussion of the trader role. When the strategy decides to withdraw it places back the Hand object in the ‘cards’ place in the same manner as the ‘bid timeout’ transition. Also the timeout value will be removed, as there is no outstanding bid anymore. The Petri-net is now enabled again to choose between the bidder (‘make bid’) and trader role (‘offer trade’).

As long as the OfferCards object is present in the ‘cards offered’ place, *offers* of other players can be rejected or accepted. This decision is delegated to the strategy interface (acceptOffer, see section 6.2.2.3 / strategy interface) by the transition ‘receive trade offer’. For this transition to be enabled there must be an incoming Offer object in the in_object place. Before querying the strategy interface it is checked if the offer is made on the bid represented by the OfferCards object. This is done by comparing the ‘bidID’ fields of the Bid objects in both composite objects. If these ids don’t match the offer will be automatically rejected by the Petri-net without querying the strategy.

It is important to note here that the ‘receive trade offer’ transition also fires if the OfferCards object represents an offer instead of a bid. In this case the incoming offer was already made invalid either by accepting another offer, withdrawing the bid or timing out the bid. For whatever reason this offer didn’t arrive at the right time (for example network delay), it needs to be rejected. One reason is that the Offer objects need to be removed from the ‘in_object’ place. A more important reason is that an offer cannot be withdrawn: the agent making this offer will wait until it gets an acceptation or rejection. For more about this see the discussion of the offer role later on.

For whatever reason it is decided to reject or accept the offer, when it is rejected the Offer object will be merged with the OfferCards object in the OfferAndOfferCards object and placed in the ‘reject’ place. The transition ‘notify trade offer rejected’ will split this object up again and send the Offer object back to the *trader*. As indicated in section 6.2.2.1 ‘bouncing’ back the Offer object will be interpreted as a rejection by the *trader*. The OfferCards object will be placed back into the ‘cards offered’ place, enabling timing out the bid, withdrawing it or accepting/rejecting other offers again.

When an offer is accepted (can only be decided by strategy interface) the merged object OfferAndOfferCards will be put in the ‘accept’ place. When the transition ‘make trade’ is fired the Cards of the OfferCards component are merged with the Offer component into a TradeCards object. The TradeCards object now represents the Offer that was accepted including the Cards of the bid this offer was made on. As can be seen in section 6.2.2.1 accepting an offer implies directly sending the cards. This is why an offer cannot be withdrawn, when the Cards are sent there is no way back. This needs a change in the communication protocol of the pit-game Petri-nets and will be (as said) left for future research.

The TradeCards object will be put in the out_object place, sending it to the *trader*. Subsequently the OfferCards component of the OfferAndOfferCards object will be put in the ‘cards sent’ place. An OfferCards object (token) in the ‘cards sent’ place means that a bid was made, an offer was accepted by sending the cards and that the Petri-net is now waiting to get the cards of the *trader* in response to this acceptation.

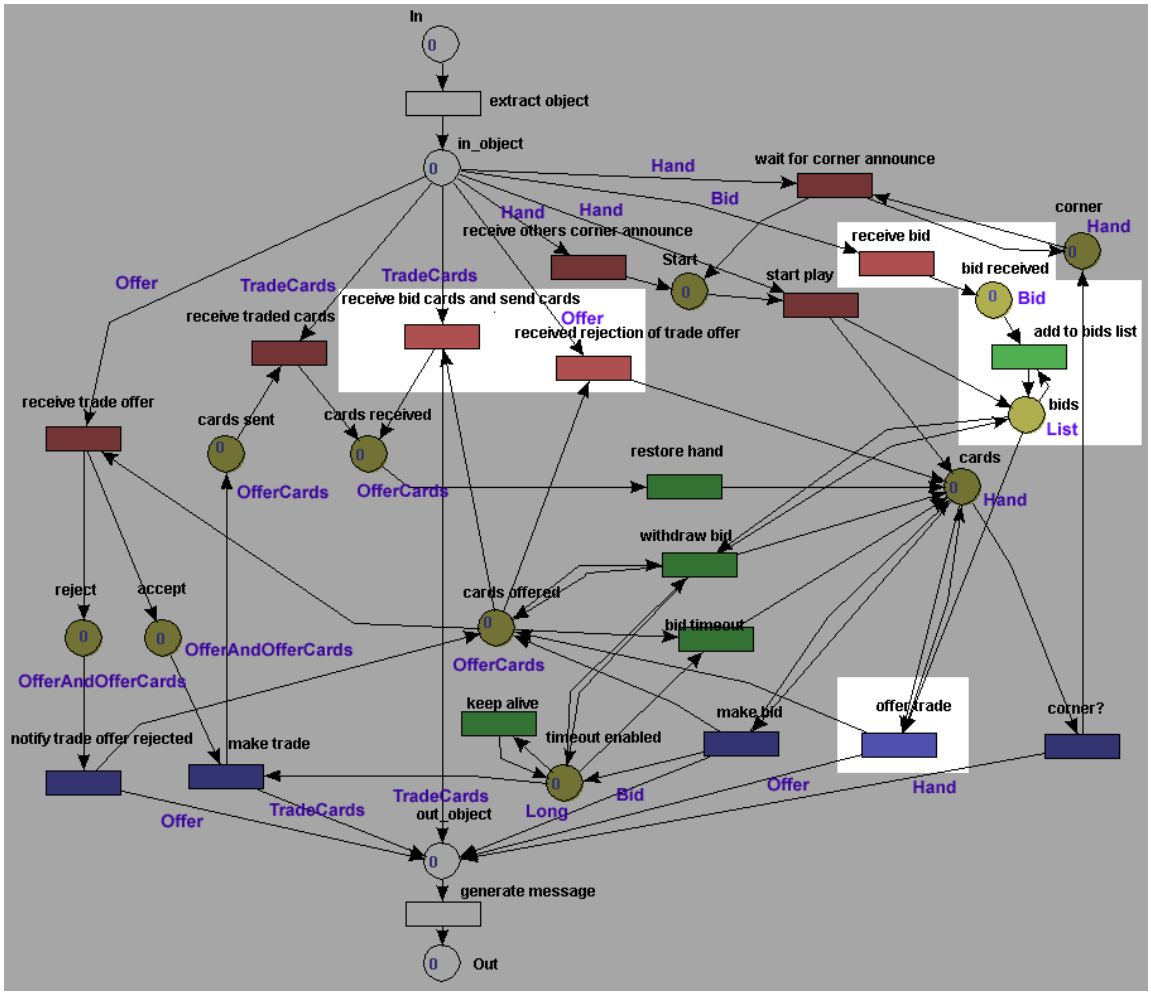
When a TradeCards object is now received in the ‘in_object’ place, it is checked if the Bid in this object and the Bid in the OfferCards object (in ‘cards sent’) have the same bidID. Note that the objects in the OfferCards objects don’t allow for checking if the received cards actually come from the same player who made the corresponding offer. It is assumed that all *player* agents use the same (non-cheating) Petri-net making the Petri-net more secure is left for future work.

Another check that is made is whether or not the right amount of cards was received. If this is not the case than the current communication protocol between the dealer and the players doesn’t have a ‘rollback’ function to solve this problem. Improving the protocol will be left for further research.

When all checks are passed the new Cards in the received TradeCards object replace the Cards object in OfferCards. When the OfferCards object is placed in the ‘cards received’ place the ‘restore hand’ transition fires. The output arc of this transition combines the cards from the Cards object and the Hand object (both form OfferCards) into a new Hand object. This Hand object is placed in the ‘cards’ place, which enables the Petri-net either to go into the *bidder* role or the *trader* role.

Trader Role

The *trader* role is performed by the following highlighted places and transitions in the Petri-net:



As said, when the ‘offer trade’ transition fires, the Petri-net enters the trader role. Before this transition can fire the Petri-net needs to have received Bid objects in its ‘in_object’ place. The transition ‘receive bid’ will place all Bid objects from the ‘in_object’ place in the ‘bid received’ place. The ‘add to bids list’ transition will take each Bid object present in ‘bid received’ one by one and add them to the List object from the ‘bids’ place.

Combining the received bids in a list is important for two reasons. When the transitions ‘offer trade’ and ‘withdraw bid’ query the strategy interface (acceptOffer & withdrawBid, see section 6.2.2.3 / strategy interface) it allows them to present an overview of available offers. Furthermore only the latest bids from each other player need to be kept, when a Bid object is received and already there exists another Bid object of the same player in the bids list the old Bid object will be replaced by the new one. Because bids of each agent get broadcasted to all the agents in the group, the amount of bids sent to each agent can become very large. By keeping only the last bid of agent the amount of Bid objects in the list cannot increase further then the amount of agents in the group.

When there is at least of Bid object in the List object contained in the place ‘bids’ and a Hand object is present in the ‘cards’ place, the ‘offer trade’ transition can fire. In the guard of the transition the strategy interface is queried for whether or not it wants to offer on one of the bids in the list (getCards, see section 6.2.2.3 / strategy interface). If

the strategy wants to offer on one of the bids, the chosen bid is returned and a Cards object representing the Cards that are selected for the *offer*. The returned cards are assumed to be abstracted from the Hand object originating form the ‘cards’ place.

The selected Bid object, the returned Cards object and the Hand object are merged into a OfferCards object which is placed in the ‘cards offered’ place. Note that the bidder field of the Bid object will indicate that this object represents an outstanding offer. An Offer object is created containing the selected Bid object plus a playerID object representing the player itself. This Offer object will be put in the ‘out_object’ place allowing it to be sent to the *bidder* of the selected bid.

As acceptation of a bid is equivalent to the *bidder* sending the cards the offer cannot be made undone, the *trader* has to wait for the *bidder* to send an acceptation or a rejection. This is a drawback of the current communication protocol which should be resolved in future work on the dealer and player Petri-nets. A possible solution to this problem is to let the dealer negotiate between two players that want to trade cards. The question is if it’s desirable to move (centralize) more functionality to the dealer Petri-net.

When the exact same Offer object sent by ‘offer trade’ is received in the ‘in_object’ place, transition ‘received rejection of trade offer’ will fire. The OfferCards object will be retrieved from the ‘cards offered’ place. The cards from its Cards and Hand object will be merged in a new Hand object which will be placed in the ‘cards’ place, allowing for a new bid or offer to be made.

When (after offering) a TradeCards object is received in the ‘in_object’ place, it will be checked if the Bid object from the TradeCards object matches the Bid object in the OfferCards object (taken from the ‘cards offered’ place). The bidID and bidder field are checked both although the bidID should be enough. Furthermore the amount of cards in the Cards object is checked with the amount of cards in the Bid object of the OfferCards object. Again, if the *bidder* didn’t send enough or too much cards nothing can be done. To resolve this problem changes in the communication protocol of the Petri-nets are needed (rollback mechanism), this will be left for further research.

When all the checks are passed the Petri-net proceeds exactly as in the last paragraph of the discussion on the *bidder* role (see previous chapter). Finally a new Hand object is made with the newly received cards, which is placed in the ‘cards’ place.

Ending & Restarting

By going through the offer and bidder role, the Hand object regularly comes back to the ‘cards’ place sometimes updated with new cards depending if a bid or offer resulted in an actual trade. Every time a Hand object is being put in the ‘cards’ place, the ‘corner?’ transition checks if the Hand object represents a corner. That is, if the cards in the Hand object are all from the same commodity.

When this is the case, it takes the Hand object from the ‘cards’ place. It then puts a copy of this object in the ‘out_object’ place and in the ‘corner’ place. The Hand object will be sent to the *dealer*, which, as explained in section 6.2.2.2, will perform some checks on the Hand object. When it qualifies as a proper corner it will be broadcasted to the other players, including this agent. Transition ‘wait for corner announce’ waits

for the Hand object contained the playerID of the agent itself, this means that the Hand object was its own. As already partially explained in section 6.2.2.1 receiving ones own Hand object in this situation means that the player has won the *hand*.

When ones own Hand is received a dummy token is placed in the 'Start' place which enabled the Petri-net to start a new game. This again is triggered by receiving ones own hand sent by the dealer, the difference is the state of the Petri-net and generally that the Hand object doesn't represent a corner. When ones own Hand is not received, it can be possible that the dealer received the Hand object (corner announcement) of another player just before the Hand object of this player.

Also in any state of the player Petri-net a corner announcement of another player can be received. Transition 'receive others corner announce' will pick up any Hand object from the 'in_object' place if the Hand object belongs to another player, which means that the Hand object is another's corner announcement (see section 6.2.2.1). After this a dummy token is placed in the 'Start' place, allowing for a new *hand* to start.

The Petri-net is designed in such a way that graphically the state of the pit-game (from the perspective of the player) can be read from the net, this means that by the presence of tokens it can be seen which transitions could (it also depends on other factors, like the strategy interface) be enabled. In the case of this transition this was impossible as it should always be enabled, for the same reason this transition cannot remove tokens from all the other players. This needs to be done in order to reset the Petri-net for the new *hand*.

This problem is solved in the 'start play' transition earlier described. In its action code a function is called which accesses the underlying net-object (part of the JFern architecture) and clears all tokens from each places. This does for sure doesn't comply with the Petri-net syntax but as the state of the Petri-net is exactly in between to games this is not a big problem. Every other solution would result in a much more complex net, which doesn't pay off in this case.

Note that now also in the case the *hand* is won by this player the whole net will be reset (that is, places will be emptied), this is done because objects from the previous *hand* can still be in the 'in_object' place. In principle this should not be the case, but the communication protocol of the (dealer and player) Petri-nets should first be perfected more. This will be left for future research.

Important Design Decisions

Behind some of the aspects of the Petri-net as specified above lie some important design decisions; these are discussed here:

- exchanging cards

The choice was made not to let the *dealer* play a role the cards exchange, the *bidder* and trader send the cards directly to each other (trades are private). If this was the case, the *dealer* could wait for both the cards of the bidder and the trader when one of both wants to trade. When the cards match up they can be swapped and sent to the appropriate agent by the dealer. When the amount of cards doesn't match, or when one of the parties refuses to trade the trade can be timed-out.

In order to represent the complete set of pit-game rules in one net, the *player* net, it was chosen to make this trading more distributed. For this a more simple protocol was used, as already partly described in the previous chapter. First of all, one of the agents, the *bidder* or the *trader*, has to be the first to send the cards. The problem is now that once the cards are sent, in principle there is no way back. This is the case because the Petri-net of the opposite party can 'look' into the cards of the other player.

When it is assumed that the Petri-net cannot be accessed by higher-level layers as the strategy object (through the strategy interface), a rollback-protocol could be made to send back the cards and make the trade undone. However, this is not implemented.

Because of time restrictions and that the pit-game agent only was needed as a test case, it was decided to keep this protocol as simple as possible. It was decided that the bidder is the first to send the cards. When a bid was made and an offer on that, in principle no more information is needed and the bidder assumes that a trade can happen directly.

This leaves the problem that because of this rule the *trader* is obliged to wait for an acceptation or rejection from the *bidder*. If something goes wrong with the acceptation or rejection message, or the *bidder* stops playing, the *trader* will wait forever. Offers can neither be withdrawn nor timed-out. Though, because the net is being reset after every *hand* the player will be able to participate again in the next game.

The other problem left open is that when something goes wrong with the sending of the cards it cannot be made undone and the only solution is to restart the current *hand*. When one player has too much cards and the other not enough the rules of the pit-game are violated.

- withdrawing bids

Players are able to withdraw their *bid* when bids of other agents have been received. In principle, timing-out a bid would be enough for a *bidder* in order to be able to make an *offer* instead of a *bid*. Though, the *player* would not be able to trade in the same way as was possible in the original card game.

First of all, in the original card game only bids are made; when the numbers of cards of two bids made by two players match, the players can trade. When the numbers don't match, they can change their bid to this number, or even bid yet another number of cards. This way players can communicate with each other in order to meet each other's demands.

Adding the withdrawing functionality enables the Petri-net players to do this communication. A player is now able to withdraw its bid for a bid of another player. When this bid is exactly the number of cards the player wants to trade, it can make an offer. The player can also decide to make a new bid with a different amount of cards in order to communicate an alternative to this (or other) players.

- returning cards

As was already pointed out, the strategy interface is expected to return cards when it decides to make an offer or bid. The main reason for this was that the pit-game Petri-net is specified to keep track of the state of the game. In order to make a clear separation, the Petri-net should not expect the strategy interface to have any memory of the state of the game. It is not expected to remember when it made a bid nor which cards it intended to bid.

Another advantage is that the strategy cannot secretly change the cards he intended to bid first. It could be argued that this doesn't make any difference, though, in the real card game the cards are held up and cannot be swapped for other cards after the decision for a trade is made.

Strategy Interface

In the explanation of the Petri-net the strategy interface was already referred to multiple times. The exact definitions of the functions will now be specified. As the name already says, it is only an interface; a strategy to be connected to this interface is needed which is defined in the next section. Please refer to section 6.2.1 for the explanation of the Java interface concept.

The strategy interface defines the following methods:

Method	**Arguments**		**Return Argument**	**Corresponding Transition**
	incoming	*State*		
getCards		Hand	cards, null=*no bid*	make bid
getCards	Bids	Hand	cards, null=*no offer*	offer trade
withdrawBid	Bids	hand, bid, cards	boolean	withdraw bid
acceptOffer	playerid	hand, bid, cards	boolean	receive trade offer
timeoutBid		hand, bid, cards, bidtime	boolean	bid timeout

The set of methods from the original micro-agent strategy interface was mainly changed so it doesn't have to keep track of any part of the pit-game state (see previous chapter: important design decisions, returning cards). The arguments passed with the methods are assumed to be enough for the strategy object to make its decision at the moment the method is called.

The transitions at the end of the table call these methods when a decision needs to be made. The arguments are split up in *incoming* and *state*. In principle all arguments come from within the Petri-net, though, the arguments specified under *incoming* are objects originating from other players outside the Petri-net while the arguments under *state* represent the state in which the *player* is. Note that the state consisting of hand, bid and cards is actually the internal OfferCards object from the player Petri-net.

In the following chapter a basic strategy implementation for this interface is defined.

Basic Strategy

Here a basic strategy is defined for initial testing purposes and later as a base for the narrative strategy implementation. The strategy also has to function as a reference strategy in the future, to signal improved performance of the narrative strategy.

Because all players in the pit-game can play autonomously and very fast the players are likely to play very chaotic, resulting in deadlocks and almost no trades being made. In this environment a narrative strategy is likelier to outperform the basic strategy. The strategy here has to be defined in a way so that players using the basic strategy play in a more synchronized way without defining complex strategies.

The basic strategy as a reference strategy should not use memory of past actions (reactive agent, see chapter 2) or anticipate on future actions. This way strategies can be compared that do this, a narrative based strategy is a good example of this. Also for this purpose, the basic strategy should function partially in a random way. When a group of players all are using the basic strategy a new strategy could win because it plays on a major weakness of the basic strategy.

Three obvious strategies exist for playing the pit-game. Trying to reach the corner first, or trying to reach a corner on the highest valued commodity, a third one is the random strategy. When trying to reach the corner first, it is best to try to corner the set of cards of which you have most and trade away as much cards at once as possible. When trying to get the highest valued corner one has to keep the cards of the highest value in any case and trade away the lower ones. At last a commodity can be chosen at random for the corner to go for at the beginning of the game and try to trade away the other cards in a random manner during the rest of the game. Trade-offs between all three strategies also exist.

The first two strategies are quite inflexible; players using these rules are more likely to reach an impasse. Especially if a player only goes for the highest cards this is the case. When choosing for the largest number of cards to keep for a corner, different players are likely to select different commodities as the corner to go for. At least not all players can have most of the same commodity. But always bidding the highest number of cards they can slow their progress down, what goes against their initial goal of trying to reach the a corner first. Bidding the highest amount could rule out some possible traders who can’t trade this number of cards and doing the same thing every time makes this even worse.

The random rules also make it less likely for different players to choose the same commodity to corner, but if this means that they go for a commodity of which they only have one, two or three cards this strategy gives a disadvantage. Having compared the three strategies, cornering the commodity of which you have the most cards proved to be the most flexible. Reassessing this every time new cars are received makes this even more flexible and helps reaching a corner faster. This also conforms better to our goal of using no memory; no memory of the chosen commodity to corner has to be made.

Randomly choosing cards to bid proved to result in the fastest trading pace; it resulted in shorter periods of impasse (periods where no cards are traded). Similarly, when choosing a bid to make an offer on, the bid is also chosen at random.

Because reaching a corner as fast as possible is beneficial in all cases the choice was made to trade as much cards as possible in the basic strategy. This is realized by always bidding when possible, always accepting offers on these bids, and making an offer whenever possible. When it comes down to withdrawing a bid for an offer caution had to be made, this will be shown later on.

The basic strategy characterized above can be used as a basis for the decisions that need to be made by each of the methods specified by the strategy interface (previous chapter):

1. **getCards** bid or not, if yes, which and how much cards to bid

As said, when the option is given, a bid is always made. The only decision to be made here then is which and how much cards to bid.

Out of the commodities that are not being cornered, a random commodity is chosen for the bid. All the cards of this commodity type are included in this bid.

2. **getCards'** offer or not, if yes, which cards to bid on which bid.

An offer is always being made when the option is given. The decision left here is which cards to offer to whom; how much cards to offer is already restricted by the bid that this offer is made on.

The offer is made on a randomly selected bid from the list of available bids. This list of bids is first purged for bids that cannot be met. When this purged list is empty no offer is and can be made. When there are bids left, cards of a random commodity are chosen for the offer providing that there are enough cards of this commodity.

Note that leaving some cards of a commodity, which is traded away in the hand, ensures that the other player can't make a corner on this commodity while trading a whole commodity gives better trading possibilities in the future.

3. **withdrawBid** withdraw current bid or not

Basically a bid is withdrawn when an offer can be made, for this please refer to the previous method.

When every player withdraws its bid every time it can make an offer and all players use this strategy, players are very likely to reach an impasse. They start off by all making a bid, when they receive each other's bids they all withdraw it. This way there are almost never any outstanding bids.

For this reason it was chosen only to withdraw with a chance of 50% in addition to the condition that an offer can be made. Also a bid has to last for at least 2 seconds so other players have time to react on the bid.

4. **acceptOffer** accept an offer on current bid or not

Offers are always accepted, the player wants to trade as much cards as possible with anyone.

5. **timeoutBid** timeout a bid or not

Each player is given a random timeout value, ranging from 3 to 15 seconds. In preliminary tests this turned out to be the best range. Giving every agent the same timeout value often resulted in agents timing out their bids at the same time and making a new bid at the same time because there were no outstanding bids.

Summary

- Two types of agents are designed, the dealer and the player.
- The choice was made for a centralized design using a dealer that broadcasts bids and corner announcements, and automatically registers players on the platform and deals/shuffles cards.
- A player requires one dealer to be present on the platform, bids and corner announcements are sent to the dealer and offers and trades are sent directly to the other players. Offers and trades will not be made public.
- The rules of the Pit game are modeled in the Petri nets of the dealer and player.
    - The player Petri net can be separated in a part that performs the bidder role and a part that performs the trader role. Which role is entered depends on whether or not the strategy object makes a bid or replies to an offer.
- Strategies that can be used by the player agents within the rules of the game are contained in strategy objects.
- A standard strategy interface is defined for connecting a strategy object to the player Petri net.
    - The strategy object is not expected to have any memory; the Petri nets represent the state of the game.
- As an example and test (reference) strategy, a basic strategy is defined which can be connected to the strategy interface.
    - In order for the basic strategy to be a (competitive) reference strategy it's designed to trade as much as possible and reach a corner commodity as fast as possible.
- Future work: make communication protocol more robust (rollback), solve security issues and extend the strategy interface.

7 Narrative Framework

Important aspects of narrative were identified in literature (chapter 4). Also agents (dealer, players) were defined with interaction rules modelling an abstraction of an e-business world (chapter 6). These agents are based on a multi-agent architecture (chapter 5). The agents are currently able to play the pit-game using a basic strategy (section 6.2.2.3 / basic strategy).

The theory will now be used to define a basic framework for the agents; this will enable them to use 'stories' as a base for more 'higher level' decisions. The current basic strategy doesn't use memory and makes decisions that don't relate to each other in any way. A narrative framework could provide them with more coherent behaviour based on their past, this will become more clear later in this chapter.

In the following chapter an overall framework will be specified in which it is tried to include the most important characteristics identified in the theory. Bordwell gives an important separation of narrative into syuzhet and fabula. The framework specified in the following chapter is a framework involving the use and construction of the fabula, for convenience the word *story* will be used instead.

The fabula will be the story model that an artificial agent uses internally; the syuzhet will be seen as what the artificial agent observes in its artificial world. What precisely characterizes a syuzhet in a multi-agent environment will be left open in this thesis. It is assumed that this syuzhet at least corresponds to a subset of the properties in Bordwell's description in section 4.5.5. This would be mainly due to the case that entertainment (for example, suspension) typically doesn't play a role within a multi-agent environment.

7.1 Overall Framework

Topic

- The theory of chapter Narrative Intelligence will be used to construct an overall framework in which separate research areas can be identified.

Here an overall framework will be given of how stories are used. First a general outline of story usage will be given. After this is done it will be described in more detail using the theory of chapter 4. Some gaps in the theory are filled in with our own arguments and other questions are left open. The attempt is made to characterize a basic narrative framework of which parts can be designed and implemented separately.

Stories need to be made when new events happen; in order to be able to find a story back it needs to be indexed in a proper way. The latter helps finding a story back at the right time so it can be applied to new situations. Applying a story can involve telling or using the story as a basis for interpretation of new events. Stories used often for interpreting new events can become culture or constitute a community if they are used by multiple individuals. Also, Schank describes several ways to combine and

manipulate stories within memory (section 4.5.3) in order to come up with more or less new stories that were not experienced ‘in reality’.

Schank does not document very well how stories are made. Bordwell describes it as a complex process of hypothesizing (see section 4.5.3). Both scientists are very brief on giving exact structures in which stories are represented. Bordwell gives a master schema as a base for his fabula (section 4.5.2); Schank specifies the index (section 4.5.1). The index seems to be a more generalized version of the fabula, which contains more specific events.

While Bordwell describes in more detail how stories are constructed, Schank describes how stories are managed in memory (section 4.5.3). The most important construct in this is his index. A story needs to be labelled well in order for it to be found at the right moment (section 4.4.2). Schank argues that the index is the story because the whole story can be used as an index. This is also the reason it needs to be more general.

In Bordwell’s theory the perceiver constructs the story as an observer; he/she doesn’t play a role in the story. In Schank’s theory stories are told and listened to, the latter process can remind the listener of a new story. In Carr’s and Dennett’s theory, but also in Schank’s, stories are seen as a way to find coherence in a person’s (or group’s) behaviour. According to Carr this is a process of anticipating future actions but also adjusting the story to what really happened (see introduction of section 4.4). How this process of anticipating works is explained in more detail by Bordwell through a mechanism of hypothesis making. The main difference seems to be that Carr takes into account that the perceiver can play a role in this story. According to Carr the perceiver is the main actor, which makes the story an autobiography.

A story is made, indexed, found back and used for telling, listening and finding coherence in one’s actions by hypothesizing about the future. The latter actually is the story-making process defined by Bordwell, which makes the story-cycle complete.

In this thesis it is assumed that future actions are decided upon because the perceiver plays a role in the story, too. Which character the perceiver plays in the story and the future hypothesis of the story will bring about what will or can happen to the perceiver. This can influence the perceiver’s actions, depending on if the perceiver wants this story to come true or not. Bordwell’s theory is tuned towards watching movies; the perceiver has no influence on the development of the syuzhet. In this thesis the agent has at least some influence on the syuzhet. The agent constructs a fabula that gives expectations about the syuzhet, the syuzhet can be influenced to make the hypotheses of the fabula come true or false; the ‘story’ is interactive.

The development of the story is not only based on hypotheses. In both Bordwell’s and Schank’s theory a basic structure is used as a starting point when a story is constructed. In Bordwell’s theory this is the template schema, in Schank’s theory this is the skeleton.

This basic structure can exist in varying complexities. Carr talks about a begin-middle-end structure which is inherent to life (see section 4.4). Dautenhahn and Bordwell talk about a more complex structure, which possibly has social origins (see

section 4.5.6). Schank's skeleton is even more specific and can be common to certain communities and cultures. This way a ranking can be given which makes these basic structures applicable to anything (life), social beings (communicating about social life) or specific cultures and communities. Leaving open the validity of this ranking, the basic structure used for story construction is called 'skeleton' from now on.

Both Bordwell and Schank mention that not always is a skeleton used; it is argued here that at least a very basic form of it is used like the begin-middle-end structure. Skeletons seem to account for many basic properties of narrative, making it a vital construct.

A skeleton provides a basic outline of the development of the story, which can give the perceiver playing a role in this story already some perspective on the future in a very early stage of story construction. This skeleton is adjusted according to what actually happened in combination with hypotheses about the future.

Skeletons are generalizations from multiple (similar) stories, which have often occurred according to Schank (see section 4.5.1). If this story occurs often within a certain group (community, culture) of individuals, this enables the group to understand each other better. Certain ways of representing (telling) events are accepted easily in a group, because individuals in this group use the same skeletons to fit these events to. According to Schank communicating is hardly possible when no skeletons are used (see section 4.5.2).

Skeletons are generalizations from old stories and are used as a basis for new stories and experiences. This way skeletons realize Schank's argument that new stories are interpreted in terms of old stories (see section 4.4.1). Skeletons provide coherence for an individual story and between stories of an individual. A skeleton gives initial coherence to a new story, and because skeletons stem form old stories the new story is likely to be structured more or less similar to the old stories.

People's actions and stories told depend on their stories in memory, which depend on the skeletons they have. This way individuals in the same environment are likely to be confronted with the same skeletons over and over again, resulting in a set of skeletons common to the people in this environment. It is argued here that culture arises implicitly through this reinforcing process.

Last but not least, skeletons can be used for indexing. In Schank's theory indices are generalized versions of stories; the same applies for skeletons. Skeletons, including some additional elements, probably play a role in indexing, making this construct even more vital to narrative.

When one is confronted with the task of constructing a new story, the question remains which skeleton to pick for the new events. The individual in restricted to the set of skeletons it has, but a choice can still be made within this set. According to Bordwell the initial set of hypotheses made when interpreting the syuzhet is very robust and not easily withdrawn. A possibility is that this initial set of hypotheses influences the selection of a skeleton; because further interpretation depends on the selected skeleton, the initial hypotheses can indeed not be easily withdrawn.

Still, in this thesis stories (skeletons) are the only concept identified as influencing interpretation of new events. Where then do these initial (and other) hypotheses come from? This still could be stories, or something else. This opens up a whole other area of research, which will be not be treated further in this thesis. A possible direction to go is given by Ashwin Rams in his paper on Interestingness [29].

Schank defines some processes of combining and elaborating stories (see section 4.5.3). As can be seen in section 4.4.2 Schank sees these processes as 'hallmarks' of intelligence.

In Schank's theory exchanging stories by telling and listening results in finding more coherence in ones stories (see section 4.4.1 / telling to oneself). Though, agents 'acting out' their stories will result in particular events happening in the syuzhet (outer world) of other agents, which will be interpreted in the form of a story.

Now some basic constructs can be specified that are needed for a basic narrative framework:

- skeleton & story structure
    - formal definition
- an indexing scheme
    - skeletons could play a role in this
- a skeleton selection algorithm
    - possibly based on Bordwell's hypothesizing scheme, initial hypotheses
- a (complex) anticipating/hypothesizing scheme
    - as characterized by Bordwell
- basic usage of a story or skeleton
    - how does a story influence the user's action
- a story generalization algorithm
    - combing multiple similar stories into a generalized skeleton

Other concepts that should be included but are not of constitutional importance to this basic framework are:

- story combination and elaboration
    - resulting in more intelligence
- explicitly telling and listening to stories
    - finding more coherence.

The skeleton and story structure combined with an indexing scheme provide the basic foundation of our framework. Stories and skeletons can be stored and found back again.

When a new story needs to be created, a skeleton selection algorithm combined with a hypothesizing scheme will be used to dynamically create a story; the story is constructed while it happens. The perceiver either plays a part in this story or not.

It is assumed here that always some basic structure, like a skeleton, is used as a starting point for the interpretation of new events. Using this structure as base, possible courses of the story can be added to the skeleton structure. These possible

courses represent a hypothesis, and can be erased or made permanent if the hypothesis is respectively made invalid or valid. Basically all the kinds of hypotheses Bordwell defines (section 4.5.3) will have to be modelled in the story using the basic skeleton as a starting point. The skeleton structure needs to be defined in such a way that it can be used for these purposes. The skeleton is transformed into a story; while the story happens, the (partial) story can be examined at any time to see which possible courses the story can take, which hypotheses are confirmed and which are 'dangling', etc.

Regardless of whether or not the story is finished or not, it can be used by the perceiver to make decisions about actions to take. For example: the perceiver finds itself in the position of the story when something terrible will happen and the perceiver plays a role in this. In this case the perceiver can act his role out, or he can refuse to do that. This depends on whether or not the perceiver chose this story as the story that he wants to happen or if he just observed it and recognized a pattern that he doesn't want to happen.

What the story exactly means for the perceiver also depends on which role of the story he assigns himself to. If the story is currently happening, determining his role is straightforward. If the story is an old one, the user of this story can put himself in the place of any 'actor' (see section 7.3.2) in the story. Because the story structure represents the (current) hypothesis of the story the story structure also needs to be defined in such a way that the current position of the perceiver (and other actors) in the story can be specified. What implications the story then has on one of his actors can be easily derived from the story structure.

In order for the basic framework to represent the whole 'life-cycle' of a story, a story generalization algorithm is needed. Multiple stories can be used to generate skeletons of any level of abstraction.

Story combination and elaboration, and listening and telling stories can be seen as processes that refine and enhance stories, skeletons and (the number of) indices within memory. However, these are not essential to complete the "life-cycle" of a story. In the design of the basic framework these concepts need to be taken into account, although first a basic framework should be created.

The various components of the basic framework are all separate areas of research. Especially the hypothesizing scheme and the generalization algorithm will be very complex research areas.

This model should not be seen as being definitive but only as a starting point to build upon and reason from. Research on the other components could result in additional requirements or more fundamental modifications to these structures. To anticipate on this some important trade-offs are discussed which were made when designing the initial model (see section 7.6).

Summary

- The following basic constructs of the narrative framework where identified: *skeleton- & story structure, indexing scheme, skeleton selection algorithm, anticipating/hypothesizing scheme, basic usage of a story or skeleton* and a *story generalization algorithm*.
- More advanced constructs are: *story combination and elaboration* and *explicitly telling and listening to stories.*

7.2 Scope of this thesis

Topic

- It will be explained what part of the narrative framework will be designed and implemented in this project.

Because of time, resource and scope restrictions it is not possible to specify the whole basic framework as outlined in the section above. For this reason the choice is made to lift out an aspect of a story's 'life-cycle'.

The processes of making stories and the generalization of skeletons from these stories are assumed to already have happened. A set of skeletons is assumed to be present. While the agent uses the skeleton it can be adjusted to what really happens, making it more flexible and making it grow into a story. Adjusting the skeleton will also prevent the skeleton, or story by then, from becoming inapplicable to the situation at hand.

However, this requires the complex hypothesizing scheme Bordwell characterizes. As an implementation of this goes far beyond the time limits of this thesis, this process is left out. In order for a 'static' skeleton to remain applicable to a sequence of events, it will need to be defined in a very generic way. In the implementation of a pit-game story in section 7.5 this will be taken into account.

Also one agent will be assumed to use one skeleton. For this reason a skeleton selection mechanism is not needed for this initial implementation. It was already pointed out that this mechanism needs more research (see section 7.1).

First a model will be defined in which stories and skeletons can be specified in section 7.3. Furthermore an idea will be given how the overall framework can be integrated into the rest of an agent's architecture in section 7.4.

Because in this thesis we will assume that an agent only uses one skeleton, this is narrowed down to how one story Petri-net can be integrated into the agent's architecture. This will still give a good idea how the whole framework can be integrated because the story Petri-net itself provides the main interface with the agent's architecture.

Finally a pit-game story will be specified in the earlier defined model in section 7.5. With section 7.4 (Using Petri-net stories) as a guideline this story will be integrated

with the rest of the pit-game player's architecture as specified in section 6.2.2.3 (New Design: Petri-Nets / Player). The latter will also provide the main link between the theoretical (chapter 4) and the more practical (chapter 6) part of this thesis.

One could argue that the design and implementation here will become somewhat trivial. In the next chapters it will become clear that already more questions are raised than can be answered in this thesis.

Summary

- A set of skeletons is assumed to be present.
- The skeleton will not be dynamically adjusted; this is why very generic skeletons will be used.
- An agent will use 1 skeleton, so no skeleton selection mechanism is needed.
- A story/skeleton model will be defined. A pit-game story will be defined in this model and integrated in the rest of the pit-game player's architecture, this way it will become evident how exactly an agent can derive its actions from a 'story'.

7.3 Defining Stories & Skeletons

Topic

- It is defined which elements are contained in a story (fabula), these elements are translated to Petri-net constructs.

In section 7.1 a story is what is constructed of the syuzhet, namely the fabula. A skeleton is a more generalized version and can be of various levels of abstraction. Because a skeleton contains less specific elements it is more easily applicable to new situations.

A story as it is perceived is in principle a linear sequence of events, it represents the order of the events exactly in the order as the happened; the sequence provides a temporal and causal relation between those events. Though, as will be seen later (in section 7.6) multiple sequences that happen simultaneously can be represented by concurrent sequences of events.

When multiple similar stories are generalized into a skeleton, similar parts of sequences are merged and dissimilar parts need to be included as 'options'. The skeleton doesn't specify exactly what happened anymore, it provides alternative ways in which the story can happen. According to Schank, also specifics like names and descriptions of physical items should be left out unless they are critical to the story [34: page 70]. In our Petri-net model this heavily depends on how the inscriptions are specified (see section 7.3.4.5).

Consider the following two (imaginary) stories:

Story 1: *"On the first day that Captain Cook walked through the desert of Australia, he fought off a crocodile by throwing stones, a day later he fought off another one in the same way."*

and

Story 2: *"When Abel Tasman walked on the North Island of New-Zealand he came across a big spider, he killed it by just stepping on it. When he later went for a stroll on the South Island he came across another one, he stepped on it again and he escaped unharmed."*

The two stories are both about men walking and coming across dangerous native animals. The persons' names are different, the animals are different and the ways to fight them off are too. A skeleton of these two stories could look like this:

Skeleton: *"When one walks and comes across crocodiles one should throw stones at it, when one comes across spiders one should just step on them."*

The two stories above only describe one order of events, namely exactly what happened. The skeleton specifies that one should be walking, but it doesn't specify how much and in which order the crocodiles and spiders will be met. Because the two stories are generalized into one, no order can be specified between killing crocodiles and spiders. Also the skeleton abstracts from the particular area that one walks in or the particular people that walked there.

Also the initial stories could be much more specific or abstract themselves, depending on how much the teller knows. The first story could have looked like this:

"On the 1st of December, 1770 Captain Cook started his walking journey along the east coast of Australia, right on the first day he met a crocodile of at least the size of the main mast of his boat the "Endeavour". Looking around the only thing he could defend himself with were stones the size of a walnuts. Remarkably these stones were so effective that one throw was enough to kill the crocodile. Up till this day no one knows which stones he used, but he killed another crocodile just a day later, just as quick using the same stones."

Stories and skeletons can be used interchangeably. The first story about Captain Cook can be seen as a skeleton of the more elaborate story above. The first story could also just as well be told in this way:

"When Captain Cook walked through the desert of Australia he came upon 2 crocodiles and fought them off both by throwing stones."

Now the order of both events is lost. This story could be generalized to a skeleton accommodating any amount of crocodiles, or if the two crocodiles are to be modelled as two specific characters the skeleton can specify two different orders in which the crocodiles can be met. As said, the question of how exactly generalizations can be made will be left for further research.

The vague line between story and skeleton leads to the belief that skeletons and stories are actually based on the same constructs. Only when a story is used as a base for interpreting new events it provides a 'skeleton' in which the new events can be placed. Still, a very specific story is too rigid to apply to any situation; a more general version is needed in order to be usable for a skeleton. Also, using any story as a skeleton defeats the purpose of skeletons as to constitute culture, assisting in communication, providing coherence etc.

Before continuing with a more formal model of the story/skeleton structure in Petri-nets first some main concepts underlying this model will be explained. For both story and skeleton the word fabula is used as they now are considered the same.

7.3.1 Key-Event

An event is a transition from one state of (part of) the world to another state of (part of) the world:

$$S_1 \longrightarrow S_2$$

The event that would follow on this event would be $S_2 \longrightarrow S_3$, resulting in a sequence; the events overlap:

$$S_1 \longrightarrow S_2 \longrightarrow S_3$$

For the fabula some events are important and other's are not. According to Bordwell (and Sengers/Bruner) the syuzhet itself defines what is an event, cause and effect; this is he calls 'narrative logic'. Though, it is arguable how much subjectivity plays a role in this selection. An event that is identified as important for a narrative (is contained in the fabula) will be called a *key-event* as opposed to events of which this key-event can consist and events that are considered as irrelevant for the story. As this chapter is about the construction of the fabula, the term key-event is used from now on instead of event.

In principle a key-event can be defined in any level of detail. For example, the key-event 'marriage' can be separated into 'wedding' and (sometimes) 'divorce'. Also a key-event can be separated by means of actors and artefacts playing a role in this. The latter depends on how the actual key-events are defined; this is further discussed in section 7.6.2.

7.3.2 Actors / Artefacts

In a narrative the collection of actors and artefacts can be called the 'world', just as the narrative defines what key-events are, it defines what the world is. Actors and artefacts that are not mentioned (or have to be assumed to be there in order for the narrative to make sense) are not part of the world in a narrative.

All actors together can be grouped as the 'cast'; the artefacts can be grouped as the 'setting'. Though the choice can be made to make this to different concepts, both have in common that (together) they represent the state of the narrative 'world'.

For this reason, in the rest of this chapter by *actors* the more general notion of actor is meant which includes artefacts. Both actors and artefacts 'act' in the world. A person can change the state of the world by starting to cry, a house can change it by collapsing.

As a key-event is a change in state of the world, it can represent a change of one actor, but also multiple actors at once. This depends on how this key-event is defined, that is, which state change the key-event represents. A key-event can consist of a sequence of state changes of its individual actors, which are (sub) key-events themselves. Note that if the order of these sub key-events is important this should be modelled explicitly

The above implies that a begin and end state of a key-event can both be defined as the state of a (sub-) set of actors.

In analogy with key-events, actors can be of various levels of detail depending on the particular story, a group of persons can act as one and a whole block of houses can collapse (earthquake). Depending on their role in the story, they can be represented as a single actor or multiple actors. A further discussion about this is included in section 7.6.1.

7.3.3 Fabula Structure

As said, one structure will be defined for both the story and the skeleton, which will be called fabula structure or fabula for short. The fabula consist of the following elements:

- Cast & Setting: Actors
- Set of Key-Events
- Temporal Restrictions on Key-Events

No functional difference is made between cast and setting, actors and key-events are defined above. The fabula consists of a set of possible key-events that could happen in any order or can have temporal restrictions on them. In the next chapter it will become clear that the Petri-nets provide a good way to model both key-events and their restrictions in a formal way.

Elements like *topic*, *goal* and *result* as defined by Schank are considered as valuable elements, especially when a skeleton or story needs to be indexed. As this concept will not be further investigated in this thesis this is also not included in the fabula structure. The focus here is on the actual arrangement of the key-events (temporal restrictions) and actors in the fabula.

The goal and the result of the story can influence whether or not an agent wants to use this story as a guideline. If the agent wants to reach the goal and the result of the story is positive (the goal is reached), the agent wants the story to become true; how an agent can influence this will become evident in the next and following chapters.

7.3.4 Stories in Petri-nets

Here a translation of the previous chapter to Petri-nets is given. In the overall framework some requirements of this model were mentioned. The Petri-net should be

flexible enough to support alternative routes within the story, usable for skeletons and hypothesizing. Though stories and skeletons can be made in any level of abstraction, the Petri-net model should provide some consistency between stories; standard rules need to be defined about how a story is made using Petri-nets. Finally the position of a particular actor in the development of the story should be modelled explicitly.

When equipping an artificial agent with story-based technology not a lot of progress is made when it only observes what is happening but keeps acting as though it didn't have this technology. The story that is made or selected needs to result in different behaviour in one or another way.

The choice is made here to let this action be derived in quite a direct way. First of all it is assumed that the current events are interpreted on the basis of a fabula that is in the process of being made or already fully completed. On the basis of this fabula one actor from the fabula will be chosen as representing the agent itself, called the *me-actor*. In section 7.3.4.2 it will become clear how the me-actor can be derived.

The fabula for a particular agent starts when it places itself in the position of a me-actor. The story ends when the me-actor token is consumed and not reproduced by a certain key-event (see section 7.3.4.4).

7.3.4.1 Tokens

Actors are represented by individual tokens, though a set of actors can be represented as a single token too (see section 7.3.2).

As the actors represent the 'state of the world', the collection of tokens in a Petri-net story can be seen as the 'world input' of the fabula. The tokens represent the state of the world, because world state changes continuously, the data in the tokens changes continuously. The Petri-net representing the fabula only registers changes important for the fabula. How this connection between 'the world' and the tokens is realized will be further discussed in section 7.4.

As a consequence, factors outside the Petri-net influence the state of the tokens and thus the functioning of the Petri-net, this could be seen as a violation of the Petri-net syntax (side-effects). The problem here is that the 'world' can behave unpredictably which cannot be defined and thus analysed anyway; only the temporal restrictions represented by the fabula as a Petri-net (see section 7.3.4.3 & 7.3.4.4) can be analysed.

Different types of actor's are represented by different data-structures, though it is possible to represent all the actors by one single type 'actor'. Representing them by different data types results in more places in the Petri-net, as for every actor/token type playing a role in a transition there has to be an input and (optionally, see last paragraph of section 7.3.4.4) an output place. This results in a better overview of which actors play in which key-events. This is in relation with the fact that actors can be modelled in various levels of detail, which is discussed in sections 7.3.2 and 7.6.1.

7.3.4.2 Transition

A transition represents a key-event; its guard represents the required state of the relevant actors in order for the key-event to happen. The required state can be

represented by any subset of the total set of actors in the fabula. The states of the actors not included in the guard are irrelevant for this key-event, since they do not play a role in it.

One of the actors present in the required state can be the *me-actor*. While the required state of other actors has to be awaited in order for a transition (key-event) guard to evaluate to 'true', the state of the me-actor can be influenced in order for the transition to be enabled. The agent, which identified itself with the me-actor, can influence his own behaviour and thus the state of the me-actor.

It is proposed here to include all required states of the different actors in the guard expression. The me-actor can for example parse the guard expression to see what it needs to do to make the key-event happen; it doesn't have to do it. Here a connection should be made which can be called the 'world output'; the state of the story can influence the state of the world in its turn (feedback). How this can be realized is discussed in section 7.4.

Note that the required states of several actors can be met at any time; there is no specified order. Should there be a certain order in which their states should change, then this should be modelled as separate key-events, because there are temporal restrictions between them. A further discussion on this matter is included in section 7.6.2.

7.3.4.3 Places

Places and arcs represent temporal restrictions between key-events. Places hold a set of actors (represented by tokens) which play a role in the key-events (transitions) having their input-arc's connected to it.

As actor's have different types for every actor type playing a role in a key-event separate input and output places are needed.

The *marking* (see section 5.2) of the net at a certain moment represents the state of the story. Note that the state of the tokens (actors) is not part of the state of the story. Their states can change continuously and not all these state changes are relevant to the story, only those that are defined in the transition guards are relevant.

7.3.4.4 Arc's

Input arc expressions represent the amount (and type, if different types in one place are allowed) of actors needed from a certain place. Usually exactly the right amount (and type) of actor's is present in the input places. When two simultaneous key-events are allowed to happen, the input arcs leading to these key-events can both take a subset of the actors from the input place. See section 7.6.1 for more about simultaneous key-events.

Output arc expressions generally reproduce all tokens that went into the transition by its input arc's. Though, the choice can be made to not reproduce actor's that don't play a role in the fabula anymore. This would represent the end of his role in the story and implies that every token in the story-net on a specific time represents an actor that still plays a role in the story.

7.3.4.5 Inscriptions

Inscriptions operate on different coloursets (types of tokens), which in our framework would be actors. Actor's can be represented by any kind of data, so inscriptions can be defined in any way. In order to make Petri-nets containing fabula exchangeable some conventions have to be made in this. In this thesis no guidelines are given for this because of time and scope restrictions of this project.

Inscriptions can be defined in a programming language (in the case of JFern, this is Java) or in a more abstract (formal) way. The latter could enable the story to be applicable in other domains as well, though, in the context of this thesis all pit-game agents can evaluate expressions in Java and they would all operate on the same kinds of data (see section 7.5.1).

Summary

- Skeletons and stories are based on the same constructs; this is why one model will be defined for them.
- The story model consists of *actors/artefacts*, a *set of key-events* and *temporal restrictions between those key-events.*
- These elements are translated to Petri-net constructs as follows:
  - Actors and artefacts are represented by tokens.
  - A transition represents a key-event.
  - Places and arcs represent temporal restrictions

7.4 Using Petri-net stories

Topic

- Before continuing with Pit-game stories it is needed to give an outline about how this story model can interact with an agent's architecture.

Tokens were identified as representing the 'world input' of the story; the guards of the transitions can result in feedback to the world, the 'world output'. However, between the world and the Petri-net story there is the agent using this story. In this chapter it will be outlined how a story Petri-net can be integrated in the architecture of an agent.

A story/skeleton only defines events that are important for the story, namely key-events. This way a story cannot define all the behaviour of an agent using this story. Though an agent can derive action from the transition guards, this is not enough for the agent to derive all its actions. The story abstracts from all events happening; it defines an abstraction of the world focussing on specific actors and key-events.

It is clear that the agent needs an underlying architecture that controls the agent. This architecture can be of any complexity and type. A story Petri-net should be integrated in this underlying architecture. This story Petri-net in its turn is part of the overall narrative framework as specified above.

Because the story Petri-net is assumed to be integrated with its underlying architecture, the Petri-net's 'world input' and 'world output' should hook into this

underlying architecture. The underlying architecture can be considered as the me-agent, though an autonomous part of the underlying architecture could be another actor. Actors that are represented by other agents can also only be accessed through the underlying architecture. What is known about the state of these actors depends on how much of it is communicated to the underlying architecture of the story Petri-net. The underlying architecture is the interface with the world.

The state of the me-actor and other actors needs to be extracted from the underlying architecture. The other way around the state of the story Petri-net, transition guards that can be enabled, should have an effect on the state of the underlying architecture. It is clear that an intermediate interface layer is needed between the story Petri-net and the underlying architecture.

How the interface layer is implemented depends on how the story layer and the underlying architecture are implemented. The interface layer can be implemented using a state table, which keeps track of the states of all the actors playing a role in the story. The state table is maintained by translating state data from the underlying architecture into this state table.

The state table needs to provide all the state variables that are used in the expressions of the transition guards. The state variables are used in the tokens and they can be altered to influence the behaviour of the underlying architecture. State variables of other actors than the me-actor are always read-only; state variables of the me-actor can be read-only too in certain cases.

If a required state for example would be to ‘see’ a certain object, this cannot be changed directly. In certain cases it would be possible to reach this state by a series of actions. It would be possible to go to a certain location where this object is. This would require additional functionality in the interface layer, which finds a way to indirectly influence the read-only state.

Though, it is assumed here that this generally will alter the meaning of the key-event. Finding this indirect way can be seen as following a sub-story, which was deliberately left out of the main story because it was irrelevant for some reason. The read-only attributes of the state variables then define exactly what can be influenced by the story and what not.

Another solution is to change ‘seeing’ a certain object into ‘looking for’ a certain object. Filling in what ‘looking for’ means, is now left to the underlying architecture. This makes defining state variables a tricky business, which is dependent on the capabilities of the underlying architecture. This way the definition of a story can depend on the underlying architecture. An abstract inscription language could be a solution here; this problem will be left for further research.

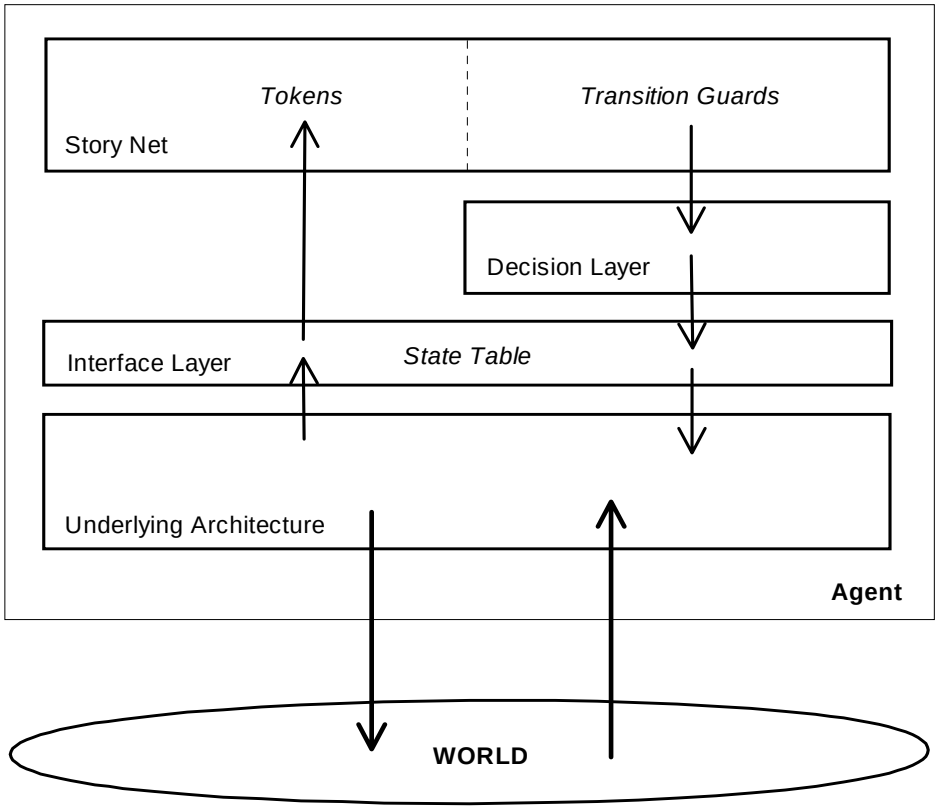
The function of the interface layer would then be determining (or only specifying) the set of state variables and their attributes. Tokens in the story Petri-net can then be references to slots in this state table, linking transitions guards to the state table is a bit more complicated.

Depending on why a story is selected, the choice can be made to realize the state of a transition guard by changing one or more state variables so the expression evaluates to 'true'. Also the choice can be made not to do this. This gets even more complex when there are multiple optional key-events that can be realized. The me-actor can make a choice between key-events depending on what course the me-actor wants the story to take.

If the story does not contain these choices and the me-actor wants every key-event in the story to happen, a one-to-one mapping can be made from the transition guards to the state table in the interface layer. Still it could be that a translation needs to be made from the language in which the guards are specified to the state table.

If the story does contain these choices, the interface layer does not only function as an interface layer but also as a decision layer. Decisions have to be made on the basis of the story. This functionality and the parsing of the guard expressions should be separated in a separate decision layer.

All the layers identified are represented in the following figure:



Note that it is not entirely clear whether or not the decision layer is part of the narrative framework or not, the decision layer at least needs to know the goal and result of the story so it can determine whether or not it should try to make the key-events in the story happen or not.

In the case of the pit-game the underlying architecture is mainly represented by the pit-game player's Petri-net and already a model was given to define stories in Petri-nets. This gives rise to the idea to also define the decision layer in a Petri-net. In this way the agents entire architecture can be separated into three levels of decisions making.

This relates back to a goal set out in the introduction (chapter 1) where we identified the benefits of separating several layers of behaviour in Petri-nets. It also fits well into research on multi-agent interaction technology at the SECML Laboratory [35].

Petri-nets in JFern can be represented in XML and can be exchanged between agents. Defining several layers of decision-making improves the exchangeability of controlling mechanisms. In the pit-game rule Petri-nets can be exchanged separate from narrative Petri-nets and other decision making Petri-nets.

In the ideal case all behavioural parts of the agent are defined in Petri-nets. When the Petri-nets all use the same inscription language the interface layer is possibly not needed anymore and the Petri-nets can be linked directly. The Petri-net representing the underlying architecture can for example be contained in the me-actor token. Also sub Petri-nets can be used, a transition can be a sub Petri-net itself. How this can be worked out further is out of the scope of this thesis and will be left for further research.

Summary

- The story model only specifies only certain key-events and actors; this is why an agent needs an underlying architecture to specify its common behaviour.
- The underlying architecture determines what is known about the world-state.
- Tokens represent this world state, and guards define a certain state of one or more tokens as being a key-event.
- The choice is made here to represent the necessary state information from the underlying architecture in a state table; this state table is directly connected to the tokens of the story-net.
- When choices need to be made on the bases of the guards of a story-net (multiple directions of the story are possible), an extra decision-layer is needed.
- In order for this decision layer to make the right decisions, Schank's *topic*, *goal* and *result* (from his index) could be included in the story model.
- All layers, except the interface layer, should be represented in Petri-nets.

7.5 Implementation

Because a story (and especially a skeleton) defined in a Petri-net can easily reach enormous proportions (see section 7.6) we define a relatively simple story in a Petri-net here. For this the model in section 7.3 is used, good examples of simple stories are those given in the introduction of this chapter.

The aim is to give an idea of what a story could be for an agent playing the pit-game and how this story can be integrated with other components of the pit-game player agent as specified in section 6.2.2.3. The previous section (7.4) is used as a guideline for this integration.

As discussed in section 7.2 the skeleton needs to be defined in a very generic way in order for this skeleton to remain applicable to a sequence of pit-game events.

7.5.1 A Pit-game story

Topic

- A Pit-game story will be defined using Petri-nets and the story model described earlier.

In the introduction of section 7.3 some examples were given in terms of relatively simple 'human' stories. The following story skeleton was given:

"When one walks and comes across crocodiles one should throw stones at it, when one comes across spiders one should just step on them."

A comparable pit-game story could look like this:

"When one has at least 3 cards of the highest commodity one should try to corner this commodity, when one is confronted with a bid of 1 card one should not offer on this bid."

In order to build a story skeleton in the form of a Petri-net from this story, the actors, key-events, and temporal restrictions in this store are needed.

First of all, it is assumed that the skeleton is a generic version of a story that shows how one *hand* in the pit-game was won by a particular player. This can be seen as the goal of the skeleton, an element that is not accommodated in our story model as story indexing and selection are out of the scope of the implementation effort of this thesis.

The actor that can be filled in for the word 'one' can be seen as the protagonist of this story[5]. If an agent identifies itself with this actor it will try to act in a way so that the story becomes true.

In addition to this there are other actors playing parts in this story, e.g. a bid is received from someone. In order not to complicate things further, it is assumed for now that there are two other players. The 'cast' of the story can then be defined as:

- me-actor: protagonist
- player 1
- player 2

[5] Note that the words story and skeleton can be used interchangeably as they were assumed to have equal structures. Previously this structure was defined as the fabula structure.

Because the story is assumed to originate from playing one *hand*, two key-events that represent the start and end of game are needed:

1. game starts
    - *guard:* me-actor received hand (a set of cards to play with).
2. game ends
    - *guard*: me-actor received corner announcement.

Initially the story seems to have two key-events, one before the comma and one after. The key-event before the comma is:

"*When one has at least 3 cards of the highest commodity one should try to corner this commodity"*.

As said the guard expression of a transition-guard represents the required state for a key-event to happen. The problem here is that the state needs to change in a particular order. First one needs 3 of the highest cards, after which the commodity to corner is changed. If these two state changes would be reversed then the same key-event would not have happened.

Because temporal restrictions are to be represented by places and not in guards in the previously defined model (section 7.3.4) the choice is made to separate this 'key-event' into two key-events. For a further discussion on this see section 7.6.2. The same applies for the 'key-event' after the comma:

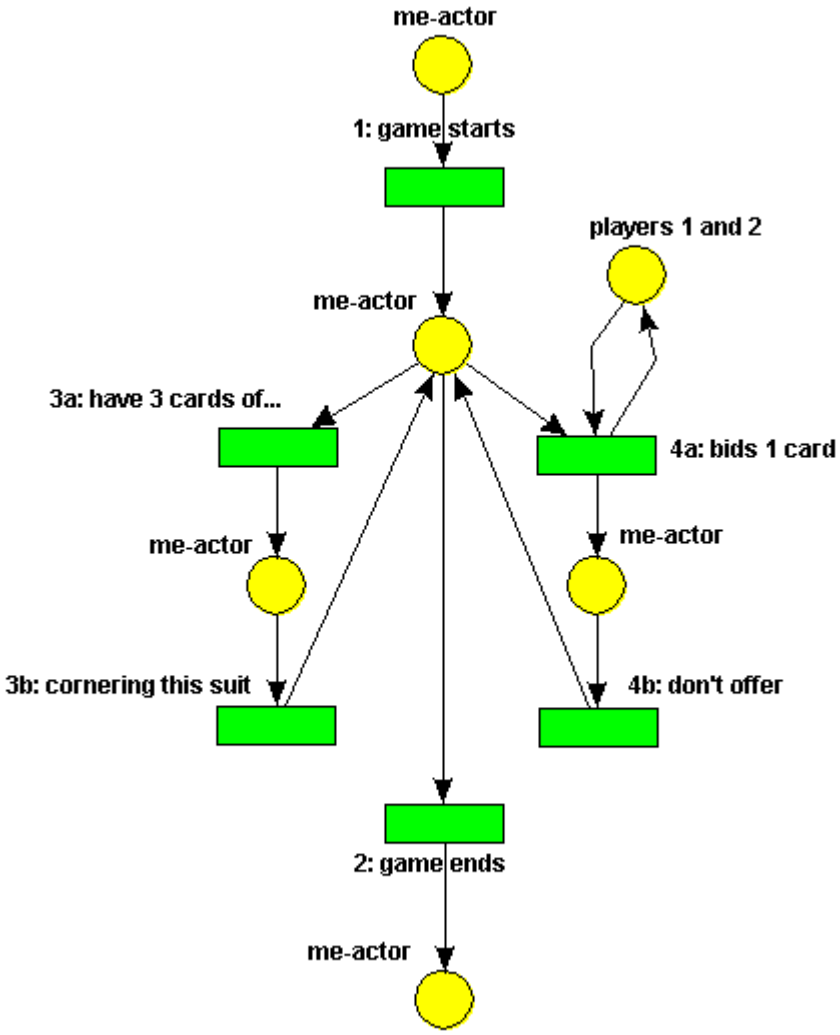
3. a) have 3 cards of the highest valued suit.
    - *guard:* me-actors hand contains at least 3 cards of the highest valued commodity.

   b) cornering this suit.
    - *guard*: me-actor's state indicates that it is trying to corner the highest valued suit.
4. a) bids 1 card
    - *guard:* player *x* (1 or 2) places a bid of 1 card.

   b) don't bid
    - *guard*: me-actor's state indicates that it won't offer on player *x*'s (1 or 2) bid.

Only the temporal restrictions still need to be defined. The a/b key-events already indicate a temporal restriction; the 'a' key-events always need to happen before the 'b' key-events. Composite key-events 3 and 4 can take place independent from each other. Before key-events 3 and 4 can occur, key-event 1 has to have happened. After key-event 2 occurred no key-events are enabled anymore.

Having defined all the essential parts of the skeleton we can now use JFern to specify a Petri-net of this skeleton:



The labels of the transitions represent the key-events modelled by these transitions. The labels of the places represent the actor type (or 'colourset' in Petri-net terms) that can be contained in those places.

As said in section 7.3.4.1 it is preferred not to define a single actor type for all actors. Two actor types are used here, the me-actor and the players that the me-actor plays against. The reason for this is that more state information is available for the me-actor, for example the cards that it has (Hand object). For the other players only by observing their behaviour some state information can be 'deduced', for example how much cards they are bidding.

The net should define some *initialisation expressions* (see section 5.25.2.2), which generate the actors for the story. Of the me-actor type there is obviously only one token, which will be placed in the place above transition/key-event 1. Of the 'other player' type there are two tokens, one for player 1 and one for player 2. These tokens will be placed in the place labelled 'player 1 and 2'.

When this Petri-net is now put into use the tokens of the me-actor and the other players will start changing state through the architecture as defined in section 7.4. Not all state changes are important for the 'story' represented in the Petri-net. Only when

the state of the me-actor changes from having no hand to having a hand, key-event 1 occurs and the story progresses (the state of the story changes).

The guards of the key-events (transitions) 1, 2, 3a, 3b and 4b only use the state of the me-actor as can been seen by their in- and output places.

The variables in the guards of key-events 1 and 2 are read-only (see section 7.4). The me-actor's state only changes when the dealer sends him a new hand or a corner announcement. Though the dealer can be represented by a separate actor the choice is made here to merge the dealer with the me-actor as the dealer doesn't really play a role in the story. Just as key-events, actors can be represented in any level of detail (see section 7.3.2).

Also the state defined in the guard of key-event 3a is read-only. The composition of the hand of the me-actor will chance as a result of its trades but cannot be changed directly.

It seems that something is overlooked when splitting up key-events 3 and 4. The Petri-net specifies that between the a/b key-events no other key-events can happen. In reality this is not the case, key-event '4a' can already happen after key-event '3a'.

In this (special) case however, key-events 3b and 4b can be realized instantly. As the goal of the story is to winning the game, the me-actor wants all key-events to happen. The guard expressions of key-events 3b and 4b also don't use state variables that are read-only. The me-actor can immediately decide not to offer on a certain agent and it can immediately change the nature of his actions as to corner the highest suit.

In this some assumptions are made about the architecture the story Petri-net is integrated in. The decision layer should, for example, immediately realize the states defined in the guard expressions. Also it depends on the underlying architecture whether or not certain states can be immediately realized. For this see the paragraph 8 to 10 of section 7.4.

The other option would be to adjust the Petri-net. In the current model this would result in a significant increase in the amount of transitions, which would negatively influence the illustrative value of this implementation. A possible solution for this is given in the sections 7.6.1 and 7.6.2.

Summary

- A generic story representing a winning scenario is defined, this means that the protagonist ('me-actor') wants to make the key-events happen. The protagonist will be the agent that uses the story.
- The story includes 6 key-events and 2 types of actors. The first type is the agent itself (the 'me-actor'); the second type represents opponent players.
- To make the Petri-net more simple and illustrative, some assumptions are made about the underlying architecture. Issues that arise when these assumptions are not made will be discussed later on.

7.5.2 An implemented story

Topic

- It will be explained how the Pit-game story as defined above fits in the Pit-game agent's architecture.

In this chapter it will be explained how the previously defined story can be implemented within the architecture of the pit-game *player* agent as specified in section 6.2.2.3. As a guideline for this the layered architecture defined in section 7.4 will be used.

The ideal implementation would be that every layer is implemented in a Petri-net for reasons already pointed out in section 7.4. Due to time limitations only the pit-game rules were designed and implemented in Petri-nets as described in section 6.2.2. These Petri-nets can be run on a specialized Opal Agents: CPN-Agents (see section 5.2.3).

As the functioning of the dealer is transparent to layers above the pit-game player Petri-net it will not be further discussed here.

The player Petri-net cannot entirely function on its own, at a certain point in the net decisions need to be made. For this a basic strategy was defined in section 6.2.2.3 / basic strategy. The CPN-Agent, player Petri-net and basic strategy together provide the underlying architecture as defined in section 7.4.

The rest of the layers were defined in Java code build upon the basic strategy, no Petri-nets were used here to speed up the implementation process. The story / skeleton of the previous chapter was defined in a Petri-net; its inscriptions were not specified formally. Because the Petri-net is very simple, the choice was made to implement the story from the previous chapter using a set of nested if-statements.

The interface layer will be implemented using a state table. To see which state variables are needed for the interface layer the guard expressions for each key-event will first be defined in (pseudo-) Java (constants are printed in italic):

Nr.	Name	Guard
1	game starts	me_actor.currenthand != *null*
3a	have 3 cards of highest suit	numcards(me_actor.currenthand, *highest*) >= 3
3b	cornering this suit	me_actor.cornerflag = = *highestsuit*
4a	bids 1 card	otherplayer.lastbid.numcards = = *1*
4b	don't offer	me_actor.boycot[x] = = *true*; x=*1..2*
2	game ends	me_actor.currenthand = = *null*

The guard expressions implicitly define some state variables. Signalling the start and end of the game is done by a state variable containing the current hand, that is, the cards the player has currently in his hand. If cards are in possession (hand is not *null*) a hand is being played, otherwise the player is waiting for new cards in order for a new game to start. It is left open where 'numcards' is specified, it is assumed to be available.

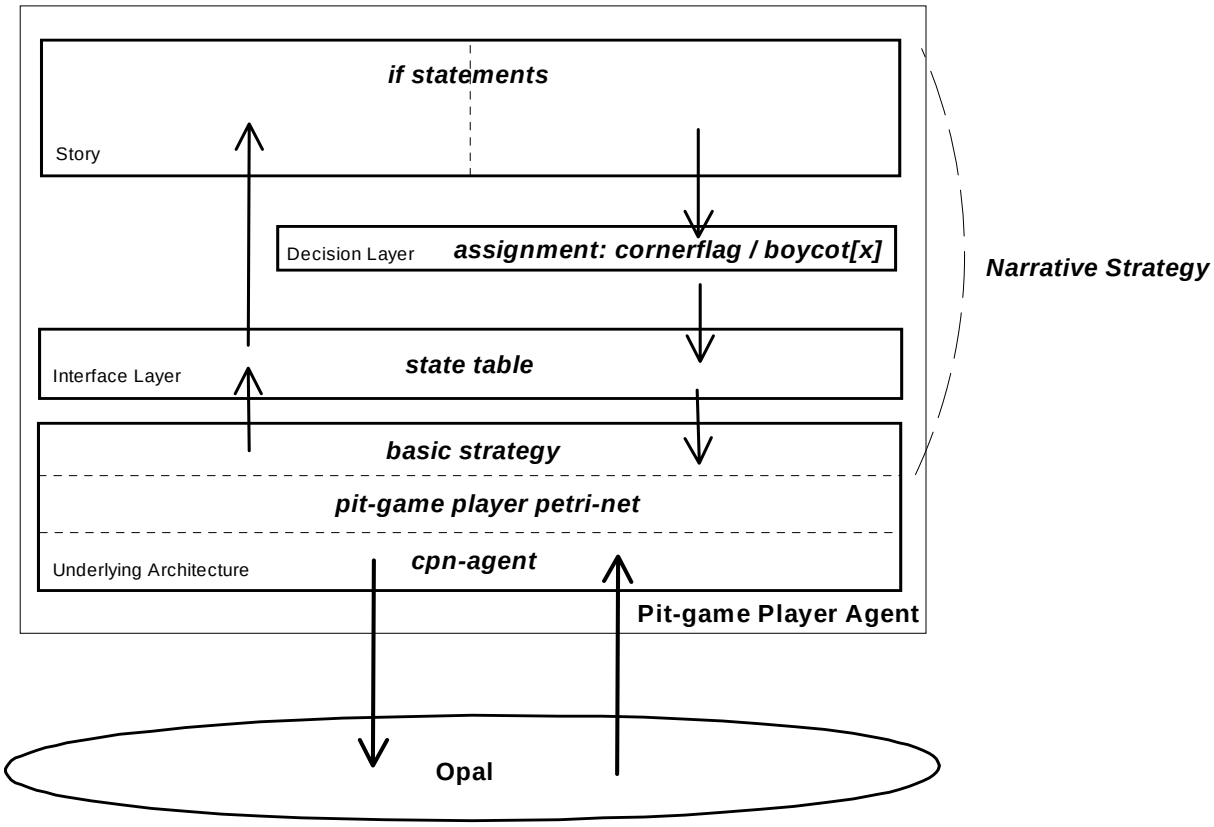
Though guard 4a only contains one variable ‘otherplayer’, two state variables are needed for this. This is due to the fact that the Petri-net (which will not be used in the implementation) would bind one of the ‘other-player tokens’ to the variable ‘otherplayer’. Because in our example two other players will be used, two state variables are needed.

The state variables ‘cornerflag’ and ‘boycot[x]’ can be set by the decision layer (see section 7.4). In the previous chapter it was argued that they are both assumed not to be read-only and that the me-actor wants all key-events to happen, provided he has influence over them. For this reason the decision layer becomes very trivial, the equalization (= = in Java) can be directly replaced by an assignment of this value to the state variable (=).

The state table of the interface layer can now be specified:

Actor	**Variable**	**Read-Only**
Player 1	player1.lastbid.numcards	x
Player 2	player2.lastbid.numcards	x
Me-actor	currenthand	x
	cornerflag	
	boycott[x]	

In the case of the pit-game player, the layers defined in section 7.4 will now look like this:



The 'hard-coded' story, state table and decision layer integrated with the basic strategy together represent the *narrative strategy*. Though it's arguable if a story is a strategy, this name is only chosen because it is connected with the 'strategy interface' as defined in section 6.2.2.3.

Summary

- Only the Pit-game rules are specified and implemented in Petri-nets. The story defined in the previous chapter is defined in Petri-nets but implemented using a set of nested if-statements.
- The CPN-Agent, the Pit-game player Petri-net and the basic strategy represent the underlying architecture. This strategy will make decisions when the story 'Petri-net' doesn't apply.
- Variables in the interface layer are specified, guard expressions are formulated in terms of these variables.
- The decision layer is trivial; when a variable in a guard expression is not read-only the value specified by the guard will be directly assigned to this state variable.

7.5.2.1 Evaluation

Topic

- Some preliminary results are discussed of testing the story-equipped Pit-game agent against Pit-game agents only using the basic strategy.

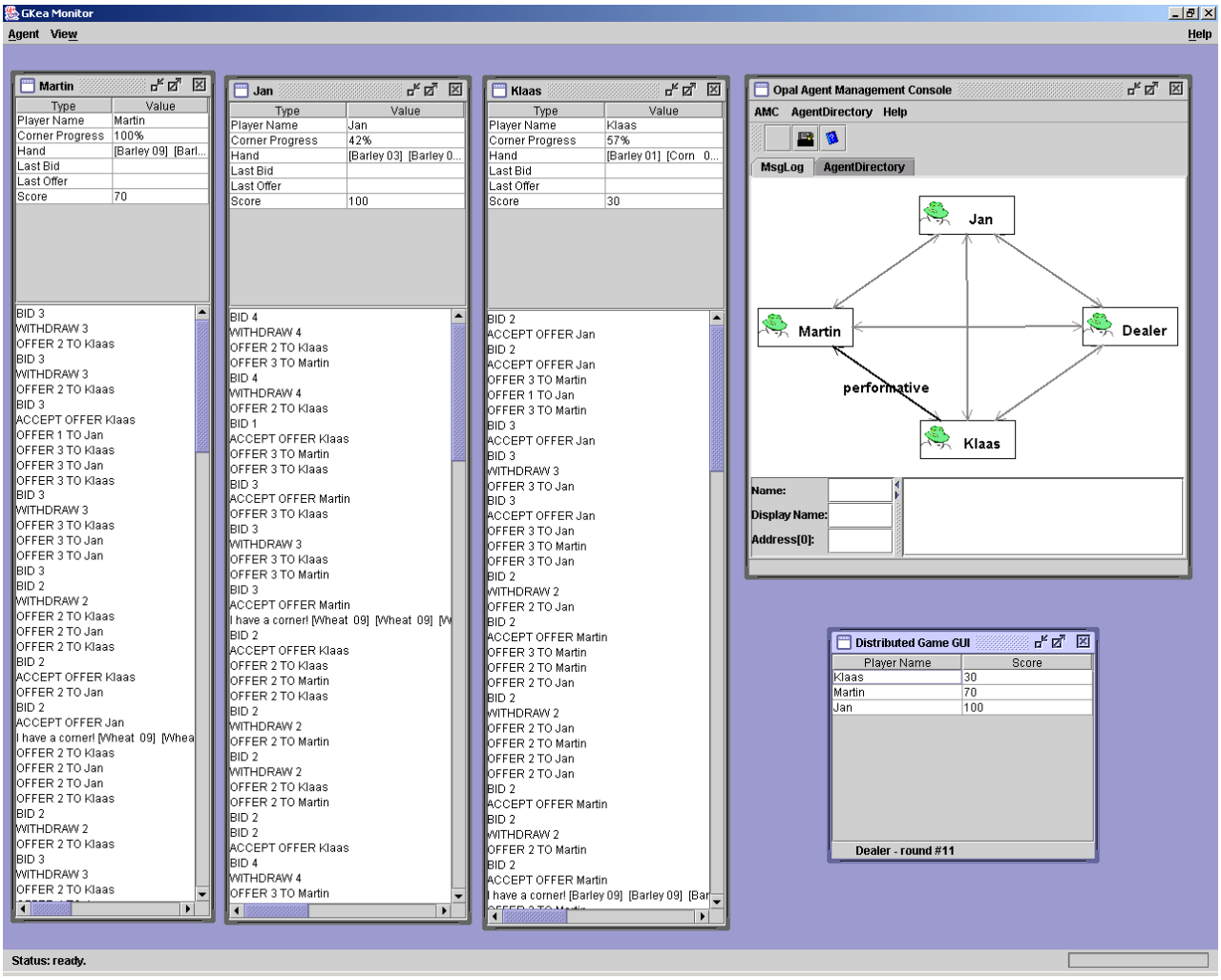
The question is now: does the behaviour of the pit-game agent improve? Several tests were performed with different numbers of agents using different, the same, or no stories. Using three player agents it turned out that they could play up to 50 *hands* after which the Opal Platform started to get more and slower. The amount of hands that they could play became only smaller when increasing the number of agents.

It is assumed here that Java's 'garbage collection' cannot clean up some objects because of an implementation error either in one of the layers of the pit-game agent as depicted above or in JFern or the Opal Platform. This way the computer's memory starts to fill up, gradually slowing the computer down. This problem surfaced at the end of the implementation period and could not be solved in the time available.

Also some unsolved problems in the pit-game protocol cause the players to get stuck in a deadlock situation. As already pointed out in section 6.2.2.3 / design decisions the current protocol used for trading is quite rigid. When messages get lost or delayed in the transport service provided by the Opal platform the player Petri-net cannot recover, for example when this causes wrong cards to be received or an offer to remain unanswered (not accepted nor rejected).

A game of 50 hands is not sufficient to produce statistically convincing test results. When running the game equipping each agent with only the basic strategy (section 6.2.2.3 / basic strategy) the relative differences between the scores of the different agents are still too big in this case. Because the agents all use the same strategy their scores should even out after a while, for this one should think more in the range of games consisting of 1000 hands. Though, to determine how many hands would be adequate more testing is needed as the main goal of this thesis (see chapter 1, Introduction) is not to give indisputable proof that narrative works better than for example the basic strategy or even better, then any strategy. Nevertheless, the behaviour of the current implementation gives some good ideas about what is needed for setting up a good testing environment for testing narrative based agents; this in addition to solving the problems with the underlying architecture.

Some tests were done with three agents, some using the story as described in the previous chapters, others only using the basic strategy. The following picture contains a screenshot of Opal running the three agents:



The three windows in the left side of the screenshot are the GUI's of each of the three *players*. In the top of these windows some information is given of the current status of the player, at the bottom a list shows all the decisions that made by this agent during the *game*. Examples of these decisions are 'BID', 'OFFER', 'ACCEPT OFFER', 'WITHDRAW', each of these have a corresponding function in the strategy interface (see section 6.2.2.3 / strategy interface). Furthermore the GUI of the *dealer* is shown at the right bottom of the screenshot, showing the current score of each player in the *game*.

The window at the right top is part of the standard Opal GUI, showing between which agents' messages were sent and between which agents currently a message is sent. In this case the agents 'Martin' and 'Klaas' are exchanging a message. These agents are currently involved in exchanging cards, as this is the only communication that can be done between players without the dealer.

When running three agents without the story they trade at a very fast pace, as already explained in section 6.2.2.3 the basic strategy was designed to reach a corner as fast as possible. When running three agents, equipping one of them with a story the 'narrative' player tends to loose more hands than the non-narrative agents. The cause of this is that the narrative as specified in the previous chapter imposes restrictions on what the agent can do, while the non-narrative agents don't have these restrictions.

Keeping the highest cards reduces the possibilities of cards the narrative player can trade. Still, keeping the highest valued cards could turn out to be more advantageous after a large number of hands. As already said, this currently cannot be statistically proven because not enough successive hands can be run.

Just as keeping the highest cards, not offering on one-card bids also reduces the amount of opportunities in which the narrative player can change cards. This is especially disadvantageous against the basic strategy. Because this strategy trades whenever it can is more likely to reach a corner before the narrative player does.

An additional problem occurs when the group of players is very small as in our test case. When the narrative player won't trade with either player the game can reach a deadlock if the narrative player owns cards such that the two other players can never reach a corner. This is because the basic strategy will stick to cornering the cards it has most from, one of the few restrictions it has.

When changing the composition of the group into two narrative players and one non-narrative player using the basic strategy a deadlock is even more likely to happen. This is because both narrative players tend to stick to the highest cards. Though, now it can be seen that the fast trading pace of the non-narrative player doesn't give it any advantage anymore. This gives rises to the idea that the particular composition of the group is very important.

When the narrative could be put out there in the real world it could perform a lot better then the agents only using the basic strategy. The amount of players in a real trading environment is typically a lot bigger in which it will probably pay to be more 'picky' because there are a lot more opportunities.

The main point here is that these small tests show that it is not that straightforward to set up a good testing environment. Summarizing, the testing results are very dependent on the following factors, which should be carefully balanced:

- Stories used by 'narrative agent'

Stories can be made in any complexity containing any information. Comparing one story or more particular stories cannot 'prove' more than that those stories in particular are beneficial for the narrative agent.

When it is needed to prove that narratives-based agents in general are more 'intelligent', whole, or at least the main part of the narrative framework as specified in chapter 7 needs to be implemented. When the agent can come up with its own stories at the right time, narrative could show its real power.

- Reference Agents

In combination with the stories used by the narrative agents, the characteristics of the other agents used to compare these narrative agents too also influences the performance of the narrative agents. While making the basic strategy an effort was made to make the basic strategy as good as possible without using any 'advanced'

strategies or techniques it could turn out that more ‘dumb’ or more intelligent agents are needed as reference agents.

- Underlying architecture of narrative agents

The way stories suggested to be used in this thesis an underlying architecture is assumed to be present; the narrative layer doesn’t provide all the functionality of the agent. Though, the most obvious is to use the same underlying architecture as that used by the ‘reference agents’.

- Number of players involved

As shown, what is the best strategy depends on the number of players. A low number of players tends to restrict the number of trading possibilities while a higher number of players allows an agent to be more ‘picky’. For a narrative agent to be able to cope with this, it at least needs to have stories for each situation and a system to select the right story at the right time (skeleton selection, see section 7.1). Even better would be to adapt the current story to make it more suitable for the current situation (hypothesis making, see section 7.1).

- The composition of the group of players

In addition to the number of players, it also matters which ‘strategies’ these players use. Again, an agent needs to adapt its story or pick the right skeleton for the right situation.

Summary

- Tests were done with three agents, 1 narrative and 2 non-narrative.
- Because of problems with java’s garbage collection and the robustness of the communication protocol between the players and dealer, only up to 50 hands could be played in a row. This is not sufficient for meaningful test results.
- The narrative agent performed worse than the agents only using the basic strategy because the story restricts the narrative agent too much in this particular test set-up.
- In order to have meaningful test-results the following factors should be carefully balanced in a test set-up: *stories used, characteristics of reference (non-narrative) agents, underlying architecture of narrative agent, number of players* and *composition of group of players.*

7.6 Model Trade-Offs

Topic

- In the previous chapter some model trade-offs were decided for. These could lead to problems when defining bigger stories; some suggestions are made to start the discussion on solving these problems.

Rules were specified about how to specify a story (fabula) or skeleton in a Petri-net. Because of these rules stories can be exchanged across different platforms and applied to different situations, though this depends on the inscription language used.

However, it was already identified that stories can be made in any arbitrary level of detail. Though the Petri-net model provides some rules for defining a story in a Petri-net, the designer has considerable freedom when specifying a story. This is especially the case when actors and key-events have to be specified. In the following chapters some guidelines are given to assist the designer.

In the design of the Petri-net model some modelling trade-offs were made which are closely related with this modelling freedom. A discussion on these trade-offs is also included in the next chapter. Note that it is not meant to be an exhaustive list; the goal here is to give a starting point for further discussion and improvement on the story model.

Because the pit-game agents are trading independently through a data transport service, no direction connection between the other players (actors) and their state can be realized. The partial state of these actors will become evident by the messages received by the me-actor. How these are linked to the corresponding tokens will not be defined here either as these are implementation dependent. Please refer to section 7.5.2 to see how this link is established in the case of this project.

Possible solutions for both problems are given in section 7.3.4.1. The tokens representing the other players can be seen as the Petri-nets used by these agents. The states of these Petri-nets can only be deducted partially through the received messages; this partial state is translated to the tokens of these players.

7.6.1 Actors

As said in section 7.3.2 all actors together represent the ‘state of the world’. The choice could be made to only use one actor token conveying the state of the world. For a simple sequential story this would not result in any problems. Still, in the graphical representation of the Petri-net in cannot be seen anymore which actor (type) has a role in which key-event; some key-events probably won’t use all of the available actors.

Also, if simultaneous and optional key-events are represented the world-state needs to be present in more than one place at the same time. Depending on how the tokens are implemented this could result in problems as multiple versions of the same state need to be synchronized continuously.

Because of both reasons it seems better to split up the world-state actor into groups of actors or even individual actors. Now more complex story lines can be modelled without having to copy actors.

Note that the model as defined in section 7.3.4 doesn’t impose any restrictions on the composition of actors. By composing several actors into one actor, transitions can transfer actor tokens even if they are not used in the key-event represented by the transition. This should be avoided in order for the graphical layout of the Petri-net to

be a better representation of the story. Arcs should show which actors play a role and do not play a role in the key-events.

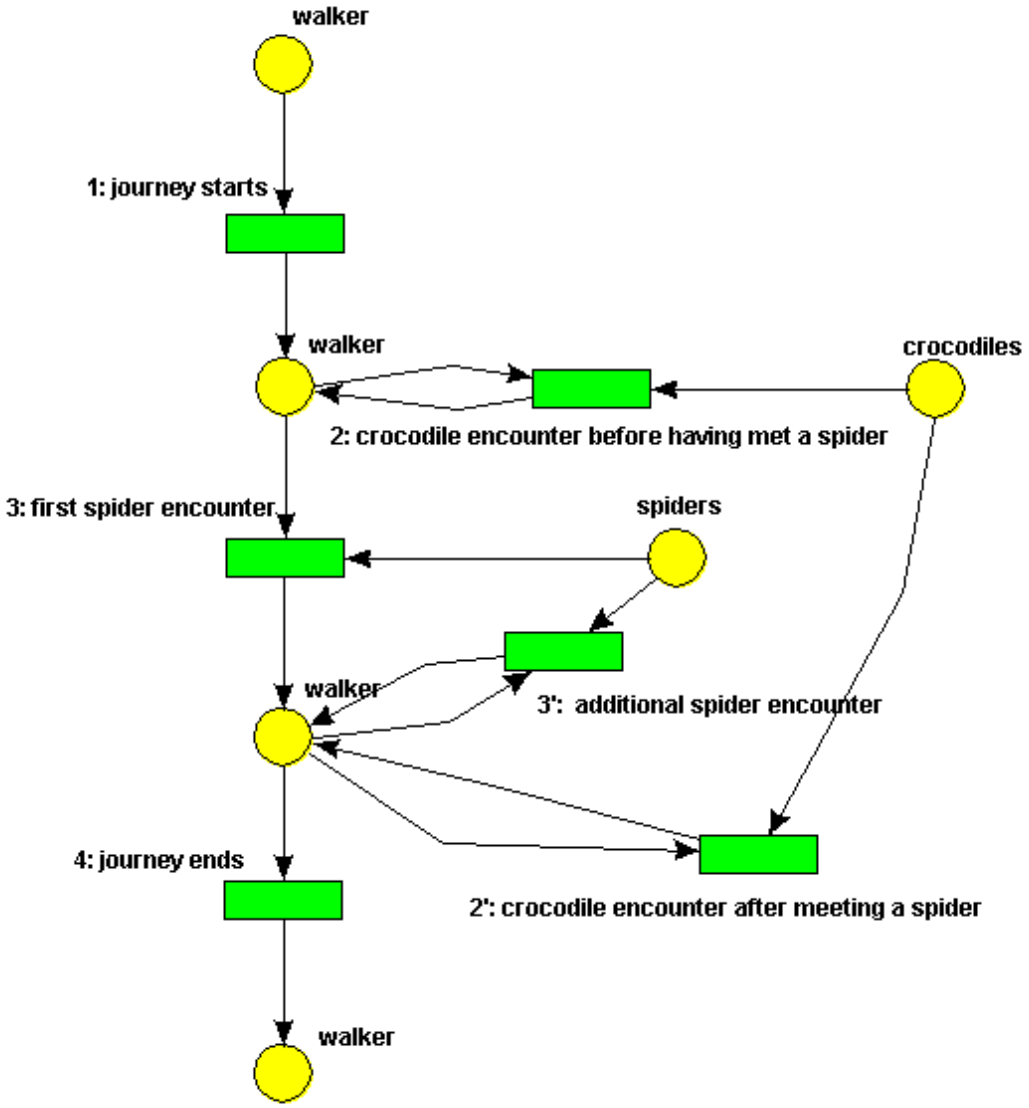
When defining a story in a Petri-net one should at least place actors in small enough groups so that all actors of this group are used in the key-events. Even smaller groups, or individual actors, can be needed to prevent actors from being copied. For example, when two detectives together play an essential role in each key-event in which one of these actors plays a role, the actors can be seen as being one actor. It doesn't make sense to add one of the suspects they have investigated to this 'composite actor'; the suspect won't play a role in subsequent key-events. When one of the detectives does something individually the composite actor will need to be split up.

Let's consider now a story representing key-events exactly in the order they happened. Some key-events can be modelled as having occurred simultaneously. Typically those key-events have a disjoint set of actors playing a role in them; an actor normally cannot participate in two key-events at once.

In skeletons optional key-events can use the same actors. The same applies when hypotheses are added to a story; two hypotheses can model to hypothesized courses of the story. Still there does not have to a problem. When the skeleton is used to map events to as they are happening (as is done in sections 7.5.1 & 7.5.2) the story only can develop itself in one way. For example, when confronted with two optional key-events using the same actors, only one of these can occur at the same time because the actors cannot participate in two key-events at once.
However, when more complex skeletons are defined problems arise when defining skeletons. Consider for example the following generalized story:

"When one walks one comes across crocodiles and spiders. It is guaranteed that one will meet at least one spider on ones walking journey."

This 'story', or skeleton, can be represented in a Petri-net in the following way:



In order to prevent actors from being copied, key-events 2 and 3 had to be distributed over two transitions. These transitions (guards) are exactly the same but the temporal restriction ‘at least’ causes both transitions to be duplicated. Also, if key-event 2 would be replaced by a whole set of sequential key-events the Petri-net would contain even more redundant transitions.

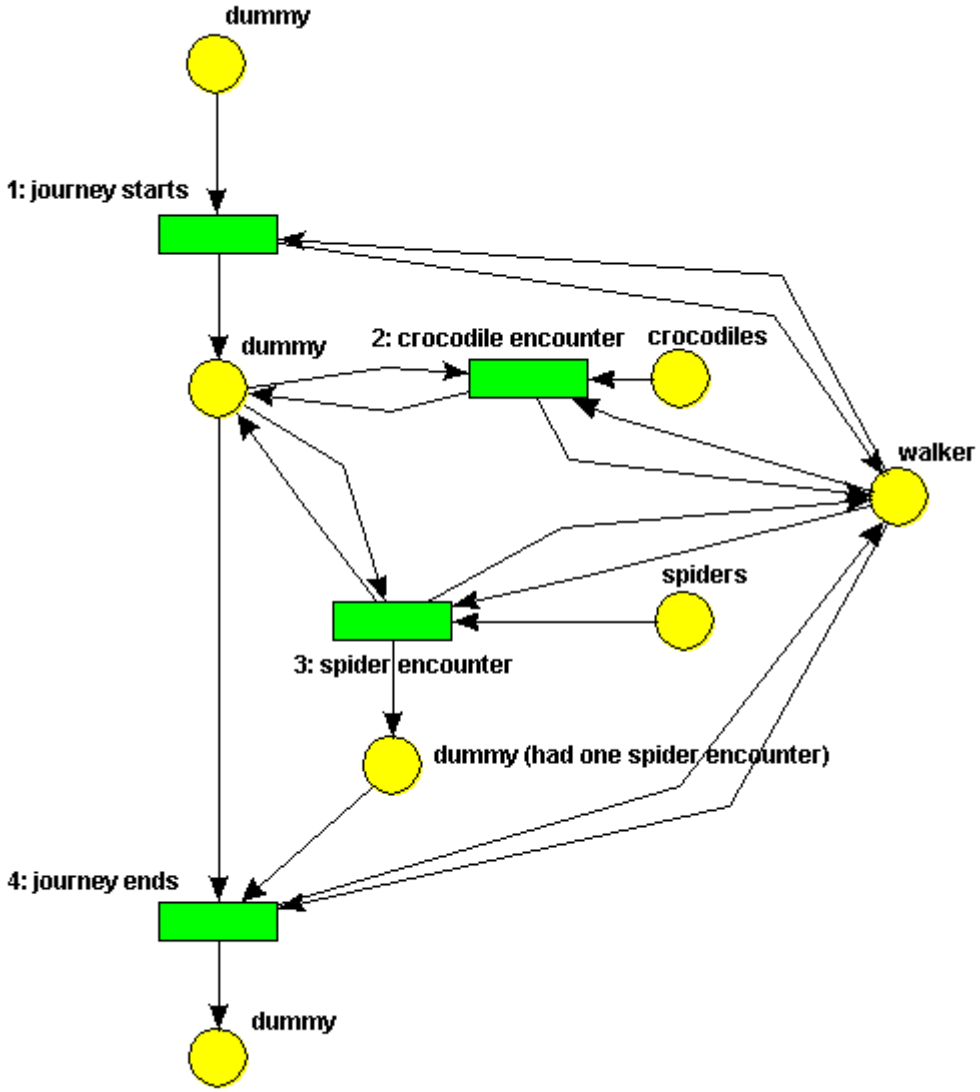
The problem here is that both the places and arcs together with the current marking (which actors are in which places) define the temporal restrictions defined by the Petri-net at a certain moment in the course of the story. In the example above the position of the walker together with the place the walker is in define which key-events can happen next, this way the walker token caries out two functions, which cannot always be combined. The state of all tokens in the net define the state of the world, the place in which they are in define the temporal restrictions in the story at a certain moment.

The only solution then to prevent key-events from having to be duplicated is to decouple the state of the world (the actor tokens in their places) from the temporal restrictions of the story (the layout of the net). This can be realized by introducing

‘dummy’ tokens which are only used to specify temporal restrictions. These dummy tokens do not contain any data so they can be generated and deleted whenever this is needed in order to impose temporal restrictions.

This also results in two types of places, places which can only contain dummy tokens and places which can only contain actor tokens. The ‘actor places’ are not used anymore for specifying temporal restrictions, all actor places and their tokens together represent the state of the world. The other places and their tokens specify temporal restrictions.

The Petri-net above would then look like this:



The three places on the right of the Petri-net (‘crocodiles’, ‘spiders’ and ‘walker’) only contain actor tokens; the four places on the left only contain dummy tokens. In order for key-event 4 to be enabled now, a dummy token is needed in the place below key-event 1 (key-event 1 needs to have happened) and a dummy token is needed in the place ‘dummy (had one spider encounter)’ (one spider encounter needs to have happened).

Because the ‘walker’ first participated in all the key-events this alternative way of modelling results in a considerable increase of arcs. Though this doesn’t always have to be the case, typically a story contains an actor performing the role of protagonist. Because this protagonist participates in most of the key-events this would result in an enormous amount of arcs to the place holding the protagonist token. A solution for this would be splitting up the story Petri-net is several ‘acts’. Then certain transitions could transfer actors to the next act, represented a new set / layer of actor-places. Because of time limits this will not be further discussed here.

When introducing acts, actor tokens in their places will again impose temporal restrictions on the story as in the model used in this thesis (section 7.3.4). The main point here is that a good trade-off needs to be found, which would depend on the stories that need to be (and can be) defined by the Petri-net model. The decisions could be made to define separate models for stories and skeletons. Though it’s hard to draw a line between stories and skeletons, skeletons typically possess more complexity. This will require more research, which cannot be done within the time limits of this thesis.

Because a more straightforward model serves just as well (or better) for illustrating the main idea of this thesis the model as defined in section 7.3.4 was used in the examples of this thesis.

7.6.2 Key-events

Key-events are events that stand out from other events because they are significant to the story. Already it was pointed out that key-events themselves can consist of sub-events (see section 7.3.1), which is due to the fact that key-events can be of any level of detail.

Some key-events consisting of multiple sub-events can be characterized by one state transition. Consider the following story for example:

“Peter’s marriage started with a beautiful wedding but ended with a horrible divorce, during his marriage Peter had several fights with his wife.”

The whole story can be summarized in one key-event: ‘marriage’. The key-event ‘marriage’ consists the sub-events ‘wedding’, ‘fight 1’, ‘fight 2’, etc., and the event “divorce”. These sub-events are not very important for the definition of the key-event ‘marriage’. The only state transition that is relevant is that from being married to not being married, the sub-event ‘divorce’. The state of being married implies that a ‘wedding’ already happened. The order and amount of fights are also irrelevant for the key-event ‘marriage’, what matters is that in the end they divorced.

Now consider this very short story:

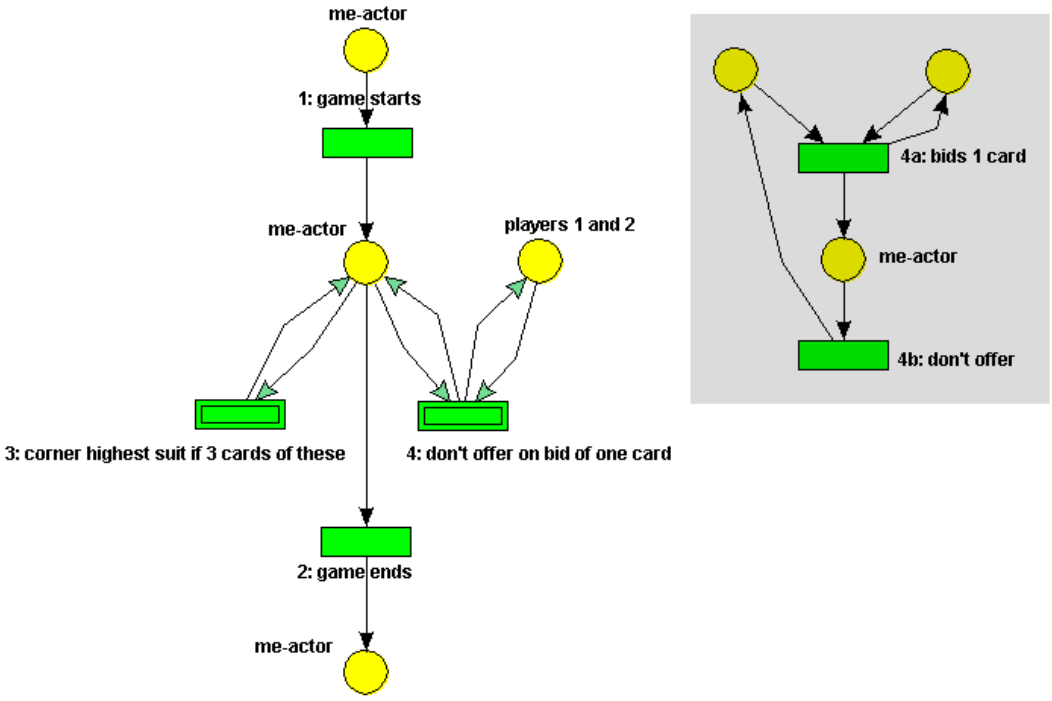
“When Captain Cook met spiders he threw stones at it.”

One could decide that the key-event should be called “fighting spiders”, because this characterized its relevance in the story. Inherent on fighting spiders however is that one needs to meet a spider first. Two state transitions need to have happened in a particular order. First from not being in the same position of a spider to being close to

this spider, ‘seeing’ the spider; second, on needs to start throwing stones. When one first starts throwing stones and then a spider comes into play, the end result is the same but the key-event doesn’t have the same meaning.

The only decision that can be made is to split this into two key-events. This decision was made in the pit-game story defined in section 7.5.1. Still, the sub key-events belong to each other, they should happen in a short time period and don’t have much meaning for the story when separated. In order to let the graphical layout of the story Petri-net represent the story or skeleton better, the decision can be made to make the Petri-net more modular. A key-event consisting of multiple sub-events, which have a particular order, can be modelled by using subnets. Subnets don’t introduce any additional rules in the Petri-net specification and are supported by JFern.

The pit-game story defined in section 7.5.1 would then look like this:



Note that this version of the pit-game story is exactly the same, only the graphical representation changed.

The subnet of key-event 4 is displayed in the grey area of the picture above. In this subnet the top left place is the me-actor place, the top right place is the place for players 1 and 2. These are ‘fusion’ places; they are exactly the same places as their corresponding places in the parent net (in the non-grey area).

This doesn’t provide a solution for the case when several sub events of key-events are allowed to happen in any order, a problem already pointed out in section 7.5.1. The me-actor plays a role in two optional key-events, in the example above key-event 3 and 4. The problem is that there are actually no temporal restrictions between the occurrences of the sub events of these key-events. The only restriction concerning both key-events is that the a-events should happen before the b-events.

If the Petri-net above was to be altered to accommodate this, it would again result in the copying of key-events (several transitions represent one key-event). Using subnets would not make a lot of sense anymore because key-events 3 and 4 would have to be integrated with each other. Using the ‘dummy token’ solution as suggested in section 7.6.1 is a good solution in order to keep key-events more modular (through subnets). For a full discussion on this please refer to this section.

Summary

- Stories, key-events and actors can be specified in any detail in the story model, some guidelines are needed for the designer.
- When it is decided which actors play a role in the story, the choice can be made to group these actors. It is suggested here that groups should be at least small enough so that all actors of a group will be used in the separate key-events of the story.
- When a story can take several optional courses, a lot of redundant transitions appear in the proposed story model. A solution for this is to introduce ‘dummy-tokens’.
- Because this solution in turn results in a lot of arcs leading from the place containing the protagonist (‘me-actor’), dividing the story into several acts could be beneficial.
- Key-events can contain several small sub events, these can be included in a sub Petri-net in order to let the visual layout of the Petri-net represent the story better.
- When sub events of several key-event can occur in any order this wouldn’t make sense anymore. Using ‘dummy-tokens’ is a good solution for this, too.

8 Conclusions

8.1 Narrative Theory

8.1.1 Orientation

In chapter 4 first of all the 'roots' of narrative were searched in order to get an idea of what the most important aspects of narrative are, this was done by combining the perspectives of several different researchers. Research from the field of (modern) Artificial Intelligence done by Sengers and Dautenhahn in section 4.3: External use was mainly used to give different perspectives on narrative. This was combined with more philosophical works of Carr and Dennet to give a deeper understanding of narrative in the introduction of section 4.4: Internal use. Using their work the basic ideas of Schank were pointed out in section 4.4.1 (Knowledge Representation).

Though initially their theories didn't seem to have much overlap, it became apparent that their theories fit together surprisingly well. The main difference was that each researcher emphasizes other aspects of narrative.

Summarizing, the following aspects of narrative seemed to be the most important:

- **coherency**: for the individual and the group.
- **culture and communities**: narratives are constitutional for the coherence of groups of any size, this relates to group coherence. Culture provides a foundation for what is normal.
- **communication**: telling stories, according to Schank telling stories improves coherence of the story.
- **balance between unexpectedness and canonicity**: a 'story' should partly conform to culture, but humans are 'tuned' to register unexpectedness.
- **language**: narrative is not dependent on language but functioned as a catalyst for the evolution of narrative.

Each researcher has a different view on the origins of narrative; this view determines on which aspect they put their emphasis on.

Dautenhahn lays emphasis the (social) communicational role of narrative, while Dennet emphasizes individual, or better, self-coherence. Through Carr's work common ground can be found between their theories. Finding coherence between an individual's past, present and future seems to be the main use of narrative. He acknowledges that narrative cannot be seen apart from its social context, this is where communities come into play. By communicating narratives, narratives can help find coherence in the group. Making coherent stories and telling stories are also important aspects in Schank's work.

It seems that the main function of narrative is finding 'coherence'. Communicating narratives lies at the basis of culture and communities. Humans are very good at constructing coherent narratives, partly they need to be challenged by unexpectedness but stories also need to conform to their culture to a certain extent.

The philosophical question of whether or not there is an ‘objective reality’ would be an interesting topic of research, especially when it is related specifically with narrative.

Throughout chapter 4 the question often arises whether or not non-narrative mechanisms influence the process of constructing narratives. In the chapters on external use this starts with the question if humans would interpret their environment the same as artificial agents. Later some concrete indications come up in Schank’s work, like ‘personal significance’ & ‘belief’ (section 4.4.1 / Behind the story) and ‘having interests’ (section 4.4.2). In section ‘What is the story?’ this can be made even more concrete with Bordwell’s ‘curiosity hypothesis’ and the fact that skeleton selection is not based on the actual events according to Schank. Due to time limitations this could not be further discussed in this thesis, however, more research needs to be done on this subject.

In section 8.3 a recommendation is made about a paper on the subject of having interests and the differences between interests of humans and artificial agents.

8.1.2 Concrete Concepts

Subsequently we looked for more concrete theory on narrative, as the goal of this thesis is to come up with more concrete definitions of narrative. In section 4.4.2 (Intelligence) already more concrete areas of research were entered, using Schank’s theory. In chapter 4.5: ‘What is the story?’ Schank’s theory in combination with Bordwell’s is used to get an idea of how the important aspects identified above can arise from specific structures (4.5.2: Fabula, 4.5.1:Index) and mechanisms (4.5.3: Making and Manipulating Stories).

The following ideas of Schank and Bordwell were most important to realize the main functions of narrative (when there are two features, the first is Schank’s version of it; the second is Bordwell’s):

- skeletons & templates
- index & fabula
- indexing stories
- combining and elaborating stories & hypothesis making
- syuzhet

The main difference between their theories lies in the fact how they make concrete how a story abstracts from ‘the world’. Schank sees it more like making generalization (index), of which in turn specific stories can be constructed again. In Bordwell’s theory it seems that the actual events have to be in the final story (fabula), reconstructing and preserving their specific causal and temporal order.

Schank’s index structure and Bordwell’s example of a template (master schema) are as concrete as it gets in their theories. Here is where this thesis enters new grounds by specifying the narrative framework. However, before this was done the pit-game agents were specified.

8.2 Pit-Game Agents

The construction of the pit-game described in chapter 6 took one third of the time available for making this thesis. This was partly due to the fact that the complexity of upgrading micro-agents to Petri-net driven agents was underestimated. Especially the development stages in which Opal and in particular JFern were in slowed down the implementation process. Often 'bugs' were found in JFern when testing the implementation of the Petri-net driven agents, it could take several days to fix a single bug.

Though, the pit-game agents are of constitutional importance to this thesis. The pit-game agents provide a test case for the narrative framework (chapter 7) and for later research. Also they show that Petri-nets can be used successfully to define, separate and exchange an agent's behaviour. Petri-nets were used in defining a story model (see section 7.3), which is part of the narrative framework. Stories defined in Petri-nets can be easily exchanged which enabled them to be used by different architectures and in different application areas.

For future research on implementing Petri-net based stories the current 'Petri-net' implementation of the pit-game combined with the story model, provide a very good point to continue from.

The dealer and player Petri-nets still need some work (see introduction of section 6.2) to make them more robust and more suitable for future testing purposes.

8.3 Narrative Framework

8.3.1 Overall Framework

The narrative framework selects the main concepts of narrative as defined in the theory and places them in relation with each other. For an exact description please refer to section 7.1, it also explained here how each concept relates to the theory.

- skeleton & story structure*
- an indexing scheme
- a skeleton selection algorithm
- a (complex) anticipating/hypothesizing scheme
- basic usage of a story or skeleton*
- a story generalization algorithm

Furthermore some advanced features were identified that are not necessarily needed for a full working system. Though, it could prove that in order for an agent to show 'real intelligence' (see section 4.4.2) these concepts will be essential:

- story combination and elaboration
- explicitly telling (as opposed to 'acting out') and listening to stories

The whole framework would constitute a complete 'narrative system', but before this can be realized a lot of research needs to be done. The concepts marked with a star were further researched in this thesis because these form the core of the narrative framework; the rest of the framework is based on the design of these concepts. The

other areas cover more research area then can be treated within the time limits in which this thesis was made.

As was pointed out in section 2.4 the original agents (section 6.2.1) needed to function in a more coherent way over time. The anticipating/hypothesizing scheme will take care of finding a coherent story, optionally based on a skeleton selected by the skeleton selection mechanism. Furthermore the story generalization algorithm will merge several similar older stories into a new story. These two processes result in stories that are coherent and also based on past experiences from the agent. This is why the agent using this story should behave more coherent and will also learn from its past experiences.

As stories can lie at the basis of a group or culture in human society, stories seem very promising in finding coherence within groups in artificial societies, too. Possibly this can be realized through the concept of explicitly telling and listening to stories (and possibly story combination and elaboration).

Story generalization and the hypothesizing scheme will probably be the most complex but also most important research areas. Both will allow an agent to 'invent' its own stories, adapting to its environment. In order to be able to make an effective hypothesizing scheme more research will need to be done on what a syuzhet exactly constitutes in an artificial world.

The indexing scheme and skeleton selection algorithm have a supportive role. Indexing was pointed out to be essential for intelligence (see section 4.4.2). Skeleton selection could be designed as part of the hypothesizing scheme. As already said, the initial hypotheses can be represented by the selection of a particular skeleton. 'Having interests' as pointed out in section 4.4.2 and 'interestingness' seem related subjects. A good paper on 'interestingness' is that of Ashwin Ram [29].

8.3.2 Design and Implementation

Some further progress was made by defining a story-model (section 7.3) together with an outline how this model can be integrated with an agent's architecture (section 7.4).

In this layered architecture it can be seen that still an underlying control mechanism is needed for the agent. Looking back at the properties of 'proactiveness' and 'reactivity' in section 2.2 and the problem of balancing those two, this architecture and the story-model form an elegant solution. The underlying control mechanism can be a reactive mechanism, making the agent flexible and responsive to its changing environment. The story-net in one of the higher layers will direct this reactive behaviour in a subtle way by only intervening at the occurrence of story-relevant events.

The story model was illustrated by a specific 'pit-game' story specified in the previously defined model in section 7.5. This 'pit-game story' was implemented on the pit-game player architecture. The story was not implemented using Petri-nets due to time limitations; this didn't pose a problem, as the example stories are very simple.

No statistically relevant test results could be obtained using this implementation. The pit-game agents need to be investigated for possible further improvements. In

particular the simple trading/bidding/offering protocol should be re-designed to make it more robust. Furthermore the ‘memory leak’ problem in the Agent Architecture (chapter 5) should be solved.

The implementation was particularly useful because it gives rise to some important factors which should be taken into account when doing further tests on narrative based agents. The following factors should be carefully balanced against each other in order to acquire meaningful test results:

- Stories used by ‘narrative agent’
- Reference Agents
- Underlying architecture of narrative agents
- Number of players involved

The initial design is meant to be a starting point for discussion on the design itself and a base for the rest of the narrative framework. Section 7.6 points out some aspects of the story model that could be done in a different way. It turned out that especially the fact that *key-events* and *actors* can be modelled in various levels of detail could lead to different variations on the current model. In this chapter it was not tried to give an exhaustive list of alternatives on the initial model, it mainly represents the point where the discussion on the story model had to be ceased due to time limitations.

8.4 End Statement

As already said in the introduction the goal of this project was to make a good case for the use of narrative based techniques in agent design. Because of the relatively young research field of Narrative Intelligence and the time limitations of this project no proof could be given that narrative improves an agent’s behaviour. Though, the theory gives many reasons to believe that narrative plays a very important role in human society and thus could also be of importance in an (artificial) agent society.

Especially the (sometimes remarkable) commonalities between research works on narrative from different research fields cause the idea that these common aspects (like coherence) must be of considerable importance. The Narrative Framework shows a concrete way how to translate this to artificial agents. As said, this Narrative Framework is only a start and we hope that this framework will lead to other researchers improving it, or at least make them think about alternatives.

9 References

1. H. Porter Abbott, *The Cambridge Introduction to Narrative*. 2002, Cambridge: Cambridge University Press.
2. David Bordwell, *Narration in the Fiction Film*. 1985, Madison, WI: University of Wisconsin Press.
3. David Carr, *Narrative and the real world: An argument for continuity*, in *Memory, Identity, Community*, L.P. Hinchman and S.K. Hinchman, Editors. 1997, State University of New York Press: Albany. p. 7-22.
4. Bruce Chatwin, *The Songlines*. re-print ed. 1987, New York: Penguin Books USA Inc.
5. Kerstin Dautenhahn. *Ants don't have Friends - Thoughts on Socially Intelligent Agents*. in *Working Notes Socially Intelligent Agents, AAAI Fall Symposium*. 1997. MIT, USA. AAAI Technical Report.
6. Kerstin Dautenhahn. *The Lemur's Tale - Story-Telling in Primates and Other Socially Intelligent Agents*. in *Narrative Intelligence*. 1999. AAAI Fall Symposium.
7. Kerstin Dautenhahn and Steven J. Coles, *Narrative Intelligence from the Bottom Up: A Computational Framework for the Study of Story-Telling in Autonomous Agents.* The Journal of Artificial Societies and Social Simulation, 2000(Starting from Society - the application of social analogies to computational systems).
8. Kerstin Dautenhahn. *The Narrative Intelligence Hypothesis: In search of the Transactional Format of Narratives in Humans and Other Social Animals*. in *Proceedings CT2001, The Fourth International Conference on Cognitive Technology: INSTRUMENTS OF MIND (CT2001)*. 2001. University of Warwick, United Kingdom: Springer Verlag.
9. Marc Davis and Michael Travers, *A Brief Overview of the Narrative Intelligence Reading Group*, in *Narrative Intelligence*, M. Mateas and P. Sengers, Editors. 2003, John Benjamins Company: Amsterdam. p. 27-38.
10. Daniel C. Dennett, *The Self as a Center of Narrative Gravity*, in *Self and Consciousness: Multiple Perspectives*, F. Kessel, P. Cole, and D. Johnson, Editors. 1992, Erlbaum: Hillsdale, NJ.
11. See for example, *Petri Nets Tool Database*, http://www.daimi.au.dk/PetriNets/tools/db.html
12. FIPA. Foundation For Intelligent Physical Agents (FIPA). *FIPA 2001 specifications*, http://www.fipa.org/specifications/
13. Martin Fleurke, Lars Ehrler, and Maryam Purvis. *JBees - An Adaptive and Distributed Agent-based Workflow System*.
14. Edwin Hutchins, *Understanding Micronesian Navigation, Hutchins*, in *Mental Models*, D. Gentner and A.L. Stevens, Editors. 1983, Lawrence Erlbaum Associates: Hillsdale, NJ. p. 191-225.
15. Parker Brothers. Parker Brothers Inc., *Pit Rules, Parker Brothers Frenzied Trading Game.*, http://www.hasbro.com/common/instruct/pit.pdf
16. AAAI Fall Symposium on Narrative Intelligence: North Falmouth, Massachusetts, http://www-2.cs.cmu.edu/~michaelm/NISchedule.html
17. Nicholas R. Jennings, Katia Sycara, and Michael Wooldridge, *A Roadmap of Agent Research and Development*, in *Autonomous Agents and Multi-Agent Systems*. 1998, Kluwer Academic Publishers: Boston. p. 275-306.

18. K. Jensen, *Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use*. 1992, Berlin: Springer-Verlag.
19. Project JXTA, http://www.jxta.org
20. M. Mateas and Phoebe Sengers. *Narrative Intelligence*. in *AAAI Fall Symposium on Narrative Intelligence*. 1999. North Falmouth, Massachusetts. AAAI Fall Symposium Series: AAAI Press.
21. Chrystopher Nehaniv and Kerstin Dautenhahn. *Semigroup Expansions for Autobiographic Agents*. in *First Annual Symposium on Algebra, Languages and Computation*. 1998. University of Aizu.
22. Mariusz Nowostawski, *JFern 3.0.0*: Dunedin, New-Zealand, http://sourceforge.net/project/showfiles.php?group_id=16338
23. Maruisz Nowostawski, *JFern Manual*. May 2002, University of Otago: Dunedin.
24. J. Odell, H.V.D. Parunak, and B. Bauer. *Extending UML for agents*. in *Agent-Oriented Information Systems Workshop (AOIS) at the 17th National conference on Artificial Intelligence*. 2000. Austin, Texas.
25. C.A. Petri, *Kommunikation mit Automaten*, in *Schriften des Rheinish-Westfalishen Institutes fur Intrumentelle Mathematik an der*. 1962, Universitat Bonn: Bonn.
26. Martin Purvis, et al. *Multi-Agent Interaction Technology for Peer-to-Peer Computing in Electronic Trading Environments*. in *Second International Workshop on Agents and Peer-to-Peer Computing, Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2003)*. 2003. Melbourne, Australia.
27. Martin Purvis, et al. *Multi-Agent Interaction Protocols for e-Business*. in *IEEE/WIC International Conference on Intelligent Agent Technology*. 2003: IEEE Press, Los Alamitos, CA.
28. Martin Purvis, et al., *OPAL: A Multi-Level Infrastructure for Agent-Oriented Software Development*. March 2002, University of Otago: Dunedin.
29. Ashwin Ram. *Knowledge Goals: A Theory of Interestingness*. in *Twelfth Annual Conference of the Cognitive Science Society*. 1990. Cambridge, MA.
30. Daniel L. Schacter, *Searching for memory: The brain, the mind, and the past*. 1996, New York: Basic Books.
31. Roger C. Schank and Robert P. Abelson, *Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures*. 1977, Hillsdale, NJ: Lawrence Erlbaum Associates.
32. Roger C. Schank, *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*. 1982, Cambridge, New York: Cambridge University Press.
33. Roger C. Schank, *Tell Me a Story: Narrative and Intelligence*. 1st ed. Rethinking theory. 1995, Evanston, IL: Northwestern University Press.
34. Roger C. Schank and Robert P. Abelson, *Knowledge and Memory: The Real Story*, in *Knowledge and Memory: The Real Story*, R.S. Wyer, Editor. 1995, Lawrence Erlbaum Associates: Hillsdale, NJ. p. 1-85.
35. SECML, *Software Engineering and Collaborative Modelling Laboratory*: Information Science Department of the University of Otago, Dunedin, New-Zealand, http://secml.otago.ac.nz/
36. Phoebe Sengers, *Anti-Boxology: Agent Design in Cultural Context*, in *Computer Science Department and Program in Literary and Cultural Theory*. 1998, Carnegie Mellon University: Pittsburgh, PA.

37. Phoebe Sengers, *Practices for Machine Culture: A Case Study of Integrating Artificial Intelligence and Cultural Theory.* Surfaces, 1999. **VIII**.
38. Phoebe Sengers, *Narrative Intelligence,* in *Human Cognition and Social Agent Technology*, K. Dautenhahn, Editor. 2000, John Benjamins Publishing Company: North Falmouth, Massachusetts.
39. David H. Uttal, *Seeing the big picture: Map use and the development of spatial cognition.* Developmental Science, 2000. **III**(3): p. 247 - 264.
40. Michael Wooldridge, *An Introduction to MultiAgent Systems*. 2002, West Sussex, England: John Wiley and Sons Ltd. 348.
41. Robert S. Wyer, *Knowledge and Memory: The Real Story*. Advances in Social Cognition. Vol. VIII. 1995, Hillsdale, NJ: Lawrence Erlbaum Associates.