Chapter 12

Comparison and Conclusions

In this final chapter, the semantic structure developed in this thesis is first compared with a number of semantic structures for other approaches. Section 12.1 presents a comparison of semantic structures for the Concurrent MetateM framework and the semantic structure developed in this thesis. To illustrate the intended use of these semantic structures, the Concurrent MetateM framework itself is described and briefly compared with the DESIRE modelling framework. Section 12.2 presents a comparison of the semantics of Object Specification Logic and the semantic structure developed in this thesis. Section 12.3 discusses the relationship between so-called local model semantics for the representation of contextual reasoning and the semantic structure developed in this thesis. Section 12.4 briefly discusses relationships with a number of other frameworks. Directions for further research are sketched in Section 12.5. Final conclusions are drawn in Section 12.6. Please note that the description of OSL in Section 12.2 is taken from (Eck, Engelfriet, Fensel, Harmelen, Venema & Willems, in press).

12.1 Concurrent MetateM

Concurrent MetateM (Fisher, 1995a) stems from research in the field of executable temporal logic (Fisher & Owens, 1995) and the MetateM programming language (Barringer, Fisher, Gabbay, Owens & Reynolds, 1996). Concurrent MetateM extends the MetateM language with constructs for concurrent programming. One of the primary applications of Concurrent MetateM is analysis and high-level executable specification of multi-agent systems. The aim and scope of Concurrent MetateM is in a number of respects comparable to the aim and scope of the DESIRE modelling framework, as is the use of temporal logic. For this reason, a relatively extensive comparison between the two frameworks and the semantic structures used to described their semantics are presented in this section. A more detailed comparison between Concurrent MetateM and DESIRE can be found in (Mulder, Treur & Fisher, 1998).

In this section, first the *two frameworks* themselves are compared. This comparison focuses on the structure of agents, inter-agent communication and meta-level reasoning. A simple example multi-agent system is then described using Concurrent MetateM and the semantic structure developed in this thesis. Finally, *semantic structures* for Concurrent MetateM are compared to the semantic structure developed in this thesis.

12.1.1 Structure of Agents, Communication and Meta-Level Reasoning

In a DESIRE specification of a multi-agent system, the agents of the multi-agent system are (usually) subcomponents of a top-level component that represents the whole (multi-agent) system, together with one or more components that represent the rest of the environment. A component that represents an agent is most often a composed component. The compositional structure of the component represents a hierarchy of agent processes. In a Concurrent MetateM model, agents are modelled as objects that have no further structure at all: all subprocesses of an agent are represented as one process, described by one set of temporal rules. In Concurrent MetateM, the environment is usually not explicitly modelled, although it is possible to introduce separate objects that represent the environment.

In DESIRE, the knowledge structures used in the knowledge bases and for the input and output interfaces of components, are defined in terms of information types, in which sort hierarchies can be defined. Information types define sets of ground atoms. Each component has an internal state, and all input and output interfaces have states. The states of components and links evolve over time. Components have the input persistence property (see Chapter 6): if, for a specific state of a component or link, a ground atom is e.g. true, than this is also the case for the next state, unless the state has changed because of updating an information link has been updated.

Concurrent MetateM does not have information types, there is no predefined set of atoms, and there are no sorts. The input and output interfaces of an object consist only of the names of predicates, called environment predicates and component predicates, respectively. Two-valued logic is used with a closed world assumption, thus a state is defined by the set of atoms that are true. Moreover, no persistency assumption is used: if an atom that is true in a specific state is not explicitly declared true in a next state, then it will be automatically false, irrespective of its previous truth value.

Communication between agents in DESIRE is defined by the information links between them. Communication between agents in Concurrent MetateM is done by broadcast message passing as follows. An agent continuously computes new truth values for its component predicate atoms (output atoms). These new values are implicitly broadcast to all other agents where they become the new truth values for the environment predicate atoms (input atoms) with the same name. Thus, when an object sends a message, it can be received by all other objects. On top of this, both multi-cast and point-to-point message passing can be defined. A newer version of DESIRE which is not discussed in Chapter 9 also provides support for broadcast communication. As an aside, let it be noted that in Concurrent MetateM, the specification of an agent may use atoms that are not part of the agent's interface. These atoms are internal atoms and are not visible for other agents.

DESIRE automatically lifts input and output atoms to a meta-level representation and provides standard information types to express statements about lifted atoms. In DESIRE, meta-reasoning is modelled by using separate components for the object and the meta-level. For example, one component can reason about the reasoning process and state of another component. Two types of interaction between object- and meta-level are distinguished: upward reflection (from object- to meta-level) and downward reflection (from meta- to object-level).

For meta-reasoning in Concurrent MetateM, the logic MML has been developed. In MML, the domain over which terms range has been extended to incorporate the names of object-level formulae. Execution of temporal formulae can be controlled by executing them by a meta-interpreter. These meta-facilities have, to the best of the author's knowledge, not been implemented in the Concurrent MetateM execution environment.

12.1.2 Example

The following example shows the specification of an example multi-agent system in Concurrent MetateM. The example also shows how the multi-agent system can be described using the semantic structure developed in this thesis.

Example 12.1. A small multi-agent system used as an example in (Fisher & Wooldridge, 1997) serves as a running example for this section. The system is described in the quote below taken from (Fisher & Wooldridge, 1997). In Figure 12.1, the system is specified in Concurrent MetateM notation (although a different notation for the temporal operators is used). The first line of each box is an agent declaration, consisting of the name of the agent, its input propositions (between parentheses), and its output propositions (between square brackets). In each box, the agent declaration is followed by Concurrent MetateM rules in temporal logic describing the behaviour of the agent. The temporal operators **F** and **Y** are to be read as 'sometime in the future' and 'at the previous moment', respectively. The following quotation from (Fisher & Wooldridge, 1997) introduces the example multi-agent system:

> "A common form of multi-agent system is based upon the idea of distributed problem solving. Here, we consider a simple abstract distributed problem solving system, in which a single agent, called *top*, broadcasts a problem to a group of problem solvers. Some of these problem solvers can solve the problem completely, and some will reply with a solution. We define such a Concurrent MetateM system in [Figure 12.1]. Here, *solvera* can solve a different problem from the one *top* poses while *solverb* can solve the desired problem, but does not annouce the fact (as solution1 is not a component predicate for *solverb*); *solverc* can solve the problem posed by *top*, and will *eventually* reply with a solution."

> “(…), we will verify some properties of the above system in [Section 6.2 of (Fisher & Wooldridge, 1997)]. We will also consider the refinement of individual agents, (e.g., a single problem-solver) into groups of agents with the same properties.”

top(solution1)[problem1,solved1]:
1. **start** → **F**problem1;
2. **Y**solution1 → solved1.

solvera(problem2)[solution2]:
1. **Y**problem2 → solution2.

solverb(problem1)[solution2]:
1. **Y**problem1 → **F**solution1.

solverc(problem1)[solution1]:
1. **Y**problem1 → **F**solution1.

Figure 12.1: A distributed problem solving system (from Fisher & Wooldridge, 1997).

To formally define the multi-agent system presented in Figure 12.1 using the semantic structure developed in this thesis, first an additional component toplevel (not included in (Fisher & Wooldridge, 1997)) is introduced. This additional component represents the multi-agent system as a whole. Communication channels in a multi-agent are explicitly represented in the semantic structure, but not in Concurrent MetateM. In Concurrent MetateM, changes to environment predicates of the agents are broadcast, so all agents are able to receive them. To represent all possible information exchange, for the four agents in the example, twelve information links are needed. However, in the example, the agents only react to environment predicates of agent top, and top is the only agent that reacts to changes to environment predicates of the other agents. Therefore, six information links suffice to represent all information transmission that takes place in the example system: one from top to each other agent, and one from solvera, solverb and solverc to top. These links are named top_to_solvera, top_to_solverb, top_to_solverc, solvera_to_top, solverb_to_top, solverc_to_top. The example multi-agent system is represented by the structure hierarchy $sh_1=\langle Comp_1;Lnk_1;\prec_1;dom_1;cdom_1\rangle$ with:

- $Comp_1=\{$toplevel,top,solvera,solverb,solverc$\}$;
- $Lnk_1=\{$top_to_solvera,top_to_solverb,top_to_solverc,solvera_to_top,solverb_to_top, solverc_to_top};
- $\prec_1=\{\langle\text{top};\text{toplevel}\rangle,\langle\text{solvera};\text{toplevel}\rangle,\langle\text{solverb};\text{toplevel}\rangle,\langle\text{solverc};\text{toplevel}\rangle\}\cup\{\langle\text{solvera\_to\_top};\text{toplevel}\rangle,\langle\text{solverb\_to\_top};\text{toplevel}\rangle,\langle\text{solverc\_to\_top};\text{toplevel}\rangle,\langle\text{top\_to\_solvera};\text{toplevel}\rangle,\langle\text{top\_to\_solverb};\text{toplevel}\rangle,\langle\text{top\_to\_solverc};\text{toplevel}\rangle\}$;

- $dom_1=\{\langle\text{solvera\_to\_top;solvera}\rangle,\langle\text{solverb\_to\_top;solverb}\rangle,\langle\text{solverc\_to\_top;solverc}\rangle,$ $\langle\text{top\_to\_solvera;top}\rangle,\langle\text{top\_to\_solverb;top}\rangle,\langle\text{top\_to\_solverc;top}\rangle\};$
- $cdom_1=\{\langle\text{top\_to\_solvera;solvera}\rangle,\langle\text{top\_to\_solverb;solverb}\rangle,\langle\text{top\_to\_solverc;solverc}\rangle,$ $\langle\text{solvera\_to\_top;top}\rangle,\langle\text{solverb\_to\_top;top}\rangle,\langle\text{solverc\_to\_top;top}\rangle\}.$

The following component information state description signatures can be used to describe the states of the components that represent the agents. (See Definition 10.12 for the definition of component information state description signatures. As this section shows how the example presented in (Fisher & Wooldridge, 1997) can be represented with the semantic structure developed in this thesis (and not with DESIRE), the DESIRE constructs presented in Chapter 9 are not used.)

CONCURRENT METATEM AGENT DECLARATION	SIGNATURE
-	$\Sigma_{toplevel}=\langle\emptyset;\emptyset;\emptyset\rangle;$
top(solution1)[problem1,solved1]:	$\Sigma_{\text{top}}=\langle\{\text{solution1}\};\emptyset;\{\text{problem1,solved1}\}\rangle;$
solvera(problem2)[solution2]:	$\Sigma_{\text{solvera}}=\langle\{\text{problem2}\};\emptyset;\{\text{solution2}\}\rangle;$
solverb(problem1)[solution2]:	$\Sigma_{\text{solverb}}=\langle\{\text{problem1}\};\{\text{solution1}\};\{\text{solution2}\}\rangle;$
solverc(problem1)[solution1]:	$\Sigma_{\text{solverc}}=\langle\{\text{problem1}\};\emptyset;\{\text{solution1}\}\rangle.$

In the example, the multi-agent system represented by toplevel does not interact with anything outside this component. Therefore, the input and output parts of $\Sigma_{toplevel}$ do not contain any proposition symbols. Moreover, in this example there is no need to define an internal part of $\Sigma_{toplevel}$,. ■

Contrary to DESIRE and the semantic structure developed in this thesis, Concurrent MetateM does not provide support for hierarchical composition. As a consequence, refinement of an agent in the model of a multi-agent system is represented by replacing the original model by its refinement. This is illustrated by a refinement of the example, again based on (Fisher & Wooldridge, 1997) to illustrate the structural aspects of components and information links. In their paper, Fisher and Wooldridge present a new system in which:

> "*solverc* is replaced by two agents who together can solve problem1, but cannot manage this individually. These agents, called *solverd* and *solvere* are defined in [Figure 12.2]. (…)Thus, when *solverd* receives the problem it cannot do anything until it has heard from *solvere*. When *solvere* receives the problem, it broadcasts the fact that it can solve part of the problem (i.e., it broadcasts solution1.2). When *solverd* sees

> this, it knows it can solve the other part of the problem and broadcasts the whole solution.” (Underlining by the author of this thesis.)

solverd(problem1,solution1.2)[solution1]:
1. (**Y**solution1.2 $\wedge$ **P**problem1) $\rightarrow$ **F**solution1.

solvere(problem1)[solution1.2]:
1. **Y**problem1 $\rightarrow$ **F**solution1.2.

Figure 12.2: Refined Problem Solving Agents (from Fisher & Wooldridge, 1997)

Fisher and Wooldridge replace *solverc* by the two agents, *solverd* and *solvere*, which together perform the task of *solverc*. Using the semantic structure developed in this thesis, *solverc* is not replaced but refined; that is: *solverc* is decomposed into two subcomponents *solverd* and *solvere*. The refined agent *solverc* is depicted in Figure 12.3.

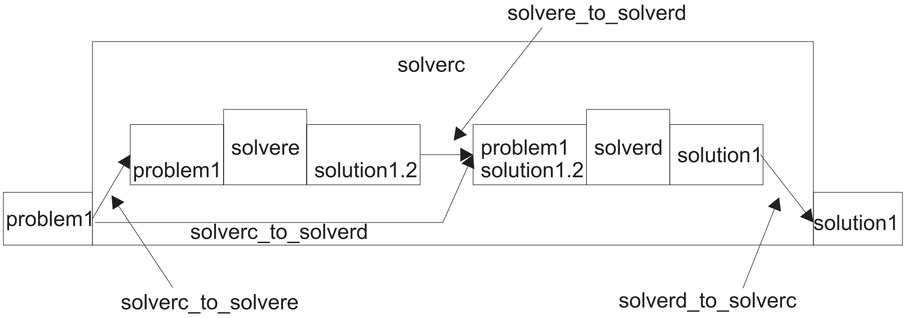

Figure 12.3: Refined Problem solving agents in graphical notation.

In Example 12.1, the compositional structure of the example multi-agent system was described by a structure hierarchy $sh_1=\langle Comp_1;Lnk_1;\prec_1;dom_1;cdom_1\rangle$. The refined system can be described by a structure hierarchy $sh_2=\langle Comp_2;Lnk_2;\prec_2;dom_2;cdom_2\rangle$ with:

- $Comp_2=Comp_1\cup\{\text{solverd,solvere}\}$;
- $Lnk_2=Lnk_1\cup\{\text{solverc\_to\_solvere,solverc\_to\_solverd,solvere\_to\_solverd,solverd\_to\_solverc}\}$;
- $\prec_2=\prec_1\cup\{\langle\text{solverd;solverc}\rangle,\langle\text{solvere;solverc}\rangle,\langle\text{solverc\_to\_solvere;solverc}\rangle,$ $\langle\text{solverc\_to\_solverd;solverc}\rangle,\langle\text{solvere\_to\_solverd;solverc}\rangle,\langle\text{solverd\_to\_solverc;solverc}\rangle\}$;
- $dom_2=dom_1\cup\{\langle\text{solverc\_to\_solvere;solverc}\rangle,\langle\text{solverc\_to\_solverd;solverc}\rangle,$ $\langle\text{solvere\_to\_solverd;solvere}\rangle,\langle\text{solverd\_to\_solverc;solverd}\rangle\}$;

- $cdom_2 = cdom_1 \cup \{\langle \text{solverc\_to\_solvere}; \text{solvere} \rangle, \langle \text{solverc\_to\_solverd}; \text{solverd} \rangle, \langle \text{solvere\_to\_solverd}; \text{solverc} \rangle, \langle \text{solverd\_to\_solverc}; \text{solverc} \rangle\}$.

For the components *solverd* and *solvere*, the following signatures are defined (the signature for *solverc* remains unchanged; it is repeated here for ease of reference):

CONCURRENT METATEM AGENT DECLARATION	SIGNATURE
solverc(problem1)[solution1]:	$\Sigma_{\text{solverc}} = \langle \{\text{problem1}\}; \emptyset; \{\text{solution1}\} \rangle$
solverd(problem1,solution1.2)[solution1]:	$\Sigma_{\text{solverd}} = \langle \{\text{problem1,solution1.2}\}; \emptyset; \{\text{solution1}\} \rangle$
solvere(problem1)[solution1.2]:	$\Sigma_{\text{solvere}} = \langle \{\text{problem1}\}; \emptyset; \{\text{solution1.2}\} \rangle$

12.1.3 Semantics

The semantics of Concurrent MetateM presented in (Fisher & Wooldridge, 1997) follows the commonly used possible-worlds semantics for modal (and, more specifically, temporal) logic. As a starting point for the semantics, the logical language available to the user for the specification of the behaviour of an individual agent (i.e., the language for the rules given in Figure 12.1) is extended with an epistemic modality. (The epistemic modality can only be applied to non-temporal formulas.) The resulting temporal logic is called TBL (Temporal Belief Logic). This logic is interpreted over time frames with a discrete order. The precise behaviour of the agents is constrained by a fixed set of temporal logic axioms. These axioms state several properties of agent interactions, e.g. an agent only believes a proposition *p* if *p* is an output proposition of another agent and *p* was true at the previous moment in time. Fisher and Wooldridge explicitly require communication to be lossless with their axiom (14) in (Fisher & Wooldridge, 1997). The order-preserving transmission property is not imposed by Fisher and Wooldridge (that is, at least not formally. However, in Fisher (1995b) it is explicitly stated that this property is *not* assumed).

At least two alternative semantics for Concurrent MetateM are reported. In (Reynolds, 1995), a first-order variant of the semantics for Concurrent MetateM is developed. In (Fisher, 1995b), a dense-time semantics is presented. The dense time semantics is basically defined in the same way as the semantics presented in (Fisher & Wooldridge, 1997). However, the frames for the interpretation of temporal formulae consist of an (infinite) set of time points with a dense order. (In fact, the set of real numbers is used.) However, the use of a dense order for the time frames has far-reaching consequences: it is not possible to express a next moment in time to interpret the 'next'-modality, or a previous moment in time to interpret the 'previous'-modality. An alternative is to focus on *changes* of truth values between states. Barringer, Kuiper and Pnueli (1986) present a dense temporal logic, called TLR (Temporal Logic of the Reals) which includes an ‘until’-modality based on state changes. The dense-time semantics for Concurrent MetateM is, in fact, a translation of Concurrent MetateM to TLR. The main assumption applied in this translation is as follows: individual agents in a multi-agent system are themselves *discrete* systems, the (discrete) behaviour of which occurs in densely ordered time, possibly simultaneously with discrete behaviour of other agents. A predicate act is introduced to represent their discrete behaviour, for each agent, which alternates between intervals in which it is true and false. Only if act is true, the truth values of other agent predicates may change. Consecutive intervals in which act is true define a ‘next’ relation for the agent.

Similar to standard temporal logic, dense time temporal logic does not have inherent support for true concurrency. However, dense-time temporal logic is viewed as a better approximation of true concurrency than discrete-time temporal logic for three reasons. First, real time is also densely ordered, so dense time frames are less artificial than discrete time frames. Second, it is possible to model concurrent actions as actions that occur infinitely close to one another. Third, and most important, in many dense-time temporal logic, all behaviour is modelled as consisting of intervals of (real) time. This enables modelling concurrent actions as overlapping intervals, possibly with the start of both intervals infinitely close to one another, and possibly likewise for the end of the intervals.

A number of differences between the semantic structure developed in this thesis and the approaches followed in the semantics of Concurrent MetateM can be distinguished:

- Both the discrete and dense time versions of the semantics of Concurrent MetateM assume that global time is available. As a result, all observers of a multi-agent system necessarily observe exactly the same behaviour in terms of the order of global states of the multi-agent system. The semantic structure developed in this thesis does not assume that global time is available. Different observers may observe different orders of global state, as explained in Chapter 7;
- Both the discrete and dense time versions of the semantics of Concurrent MetateM describe the behaviour of a multi-agent system in terms of global states of the entire multi-agent system. There is no support for locality;
- In Concurrent MetateM, only one view is provided, which completely describes the behaviour of a multi-agent system. Thus, this view is comparable to the glass box view. The semantic structure developed in this thesis provides three views on the behaviour of a multi-agent system;
- In the semantic structure developed in this thesis, the behaviour of compositions of components is described in terms of the behaviour of these components by incorporating the behaviour of these components in compatible multitraces. Multitraces retain the hierarchical structure of components. In Concurrent MetateM, the behaviour of compositions of

components is described by models of a global set of formulae, composed of the formulae that describe the behaviour of individual objects. Moreover, objects in Concurrent MetateM are not hierarchical, i.e., objects cannot consist of other objects.

12.2 Object Specification Logic (OSL)

Object Specification Logic (OSL, Sernadas, Sernadas & Costa, 1995) is a logic designed for the specification and analysis of object-oriented models of information systems. OSL is of interest for this thesis because separation of local and global reasoning has been an important requirement in the design of OSL. As a result, OSL supports locality, and, to a limited extent, compositionality. Locality and compositionality are important features of the semantic structure developed in this thesis. Moreover, OSL is state-based, as is the semantic structure developed in this thesis, and employs temporal logic, as does DESIRE. The description of OSL in this chapter is taken from (Eck, Engelfriet, Fensel, Harmelen, Venema & Willems, in press).

OSL consists of two levels: a local and a global level. The local level is concerned with the definition of the local state and behaviour of an object (the description of which is called an *aspect* of an object), specified by *aspect templates* in a local specification language. (For each aspect template, a different local language is defined.) At the global level, the different aspects are related, forming specialisations (to represent inheritance) and aggregations (to represent composite objects). In this section, the notation and terminology used in (Jungclaus, 1993) is adopted, which differs from the (more complex) notation in (Sernadas *et al.*, 1995). Section 12.2.1 presents the local specification language, used to specify object aspects. Section 12.2.2 describes the semantics of the local level. Section 12.2.3 proceeds to discuss morphisms, which connect the local and the global level. Section 12.2.4 concludes the description of OSL by discussing the global level and its semantics. Section 12.2.5 presents a brief summary of the preceding sections on OSL. Section 12.2.6 compares the semantics of OSL to the semantic structure developed in this thesis.

12.2.1 Local Syntax

At the local level, states and transitions are described using many-sorted first-order local temporal predicate languages. These languages are defined using (local) signatures $\Theta=\langle \mathcal{IS};\alpha;ATT;EVT\rangle$. A local signature consists of a partially ordered set[4] $\mathcal{IS}$ of identifier sorts (sorts containing object identifiers), a distinguished sort $\alpha\in\mathcal{IS}$ (the local sort), which is used for identifying instances of the class (aspect)

---

[4] Actually, $\mathcal{IS}$ is the Cartesian category freely generated from the partial order. In this section, OSL is presented without reference to the category structure of the sorts, similar to the description in (Jungclaus, 1993).

specified by Θ, a set of attribute symbols ATT and a set of event symbols EVT. Based on a signature Θ, a set of predicates $\mathcal{P}(\Theta)$ is defined, consisting of predicates $\triangleleft e(x_1,\dots,x_n)$ and $\otimes e(x_1,\dots,x_n)$ for every event symbol $e \in EVT$, and $\triangleleft a(x_1,\dots,x_n)$ for every attribute symbol $a \in ATT$. The intended meaning of the predicates $\triangleleft e(x_1,\dots,x_n)$ and $\triangleleft a(x_1,\dots,x_n)$ is that event e is enabled (i.e., *can* occur (it does not have to occur) in the current state) and that a certain value for attribute a is observable. The predicate $\otimes e(x_1,\dots,x_n)$ indicates that event e occurs in the current state. From these predicates, a local specification language is defined, with the usual logical connectives and three temporal operators: $\bigcirc$ ('next'), $\Box$ ('always') and $\Diamond$ ('sometimes'). In this local language, predicates are localised by prefixing them with a variable with as sort the sort α. A *local specification* is a set of formulas (the local axioms) in this language. State transitions are described by axioms that define admissible behaviour by relating current and future states.

12.2.2 Local Semantics

Similar to the syntax of OSL, semantically local and global states are distinguished. The local interpretation of formulas assumes the existence of a fixed data universe (an algebra of data types), providing a carrier set $A(s)$ for each sort s. Formulas in the local specification language are interpreted over local interpretation structures that consist of a family of carrier sets for the sorts used and a sequence of states $(\sigma_k)_{k \in N}$ (discrete linear time). States are sets of predicates that describe which observations are possible, which events are enabled, and which events actually occur in that state. Formally, states are elements of the set of all possible local states $\Pi = \{p(x_1,\dots,x_n) \mid p \in \mathcal{P}(\Theta), x_i \in A(s_i) \text{ for } i=1,\dots,n\}$. A sequence of states has additional semantic constraints, among which is the following frame assumption: only the occurrence of an event can change the set of observables and enabled events (i.e., only an event occurrence can change a state). Another constraint is that attributes are functional in OSL: if $\triangleleft a(x) \in \sigma$ and $\triangleleft a(x') \in \sigma$ for some state σ and for some $a \in ATT$, then $x=x'$. As a consequence, attributes are interpreted functionally, as usual.

12.2.3 Template Morphism Syntax: Bridge Between Local and Global Level

As stated before, objects can be composed to form complex objects. In OSL, this is specified using *template morphisms* to compose complex signatures from which formulae describing the complex objects are generated. A template morphism σ: $\Theta \to \Theta'$ between two signatures $\Theta = \langle \mathcal{IS}; \alpha; ATT; EVT \rangle$ and $\Theta' = \langle \mathcal{IS}'; \alpha'; ATT'; EVT' \rangle$, is a tuple $\langle \sigma_{IS}; \omega_\alpha; \sigma_{ATT}; \sigma_{EVT} \rangle$, with σ_{IS} a mapping of identifier sorts, σ_{ATT} and σ_{EVT} mappings for the attribute and event symbols, respectively, and ω_α: $\alpha' \to \sigma_{IS}(\alpha)$ an operator used to distinguish kinds of morphisms in a way that is not relevant here. There are two kinds of morphisms: *inclusion* of Θ in Θ' and *injection* of Θ in Θ'. With the former, additional signature elements can be added to Θ, making inclusion morphisms suitable for modelling specialization. With the latter, Θ can be incorporated in a more complex signature, making it suitable to model aggregations of objects in composed objects. Using template morphisms, different local signatures can be combined. The resulting signatures are used to compose formulae in the global language.

12.2.4 Global Language and Semantics

The global language is based on a signature that consists of all local signatures, generated by inclusion and injection template morphisms. The language defined over this signature consists of formulas of the form $\varphi \Rightarrow \psi$, where φ and ψ are local-language interaction formulas over different signatures. Using this language, relations between (the behaviour of) objects are described. In a formula $\varphi \Rightarrow \psi$, the occurrence of events described in φ implies the simultaneous occurrence of the events described in ψ ('event calling'). The semantics of the global language is similar to the semantics of local languages. (The global language is generated from the global signature in the same way as local languages are generated from local signatures. The semantics of the global language can therefore be defined in the same way as the semantics of the local language.) The global language is thus interpreted over sequences of states, each of which is a set of predicates of the global signature. These states are global states.

There is, however, one difference between the global level and the local level. The global signature consists of local signatures, related by template morphisms. The semantics of the global language not only consists of sequences of states generated from the global signature, but also of an indexed set of all local interpretation signatures. Similar to the semantic structure developed in this thesis, only combinations of local and global interpretation structures that respect specific constraints are allowed. (See (Sernadas, Sernadas & Costa, 1995, Definition 5.4, second bullet, clause 2, page 621. In this thesis, these constraints are modelled by compatibility relations.) However, in OSL, local interpretation structures are only related to the global interpretation structure, which needs to be defined to interpret formulae in the global language. In the semantic structure developed in this thesis, compatibility relates local component and link traces, and no global traces are necessary.

12.2.5 Intuition Behind OSL

Consider a global language formula $\varphi \Rightarrow \psi$. Both φ and ψ are formulae of the global signature, containing subformulae that refer to different objects in the system. At the global level, with formulae like $\varphi \Rightarrow \psi$, it is only possible to formulate expressions like 'if something happens in the life of an object referred to in φ, then something else has to happen in the life of an object referred to in ψ at the same moment'. For such an expression to have a meaning, it is necessary to specify the lives of both objects. This is done by the full temporal logic subformulae of φ and ψ.

12.2.6 Comparison

The semantics of OSL resembles the semantic structure developed in this thesis in its attention for locality. As in the semantic structure developed in this thesis, OSL starts from local perspectives on the behaviour of parts of a system (objects in the case of OSL, components in the case of the semantic structure developed in this thesis). A number of differences, however, can be identified:

- OSL constructs a global language to describe the behaviour of a system as a whole, which is not the case for the semantic structure developed in this thesis. (As OSL is a language, the only way to relate different local languages is to define a global language.)
- OSL supports compositionality only to a limited extent. In OSL, it is possible to specify objects that consist of other objects (aggregation). However, the behaviour of a composite object is not described in terms of the behaviour of its constituent objects.
- In OSL, only two levels of locality are distinguished: the object level and the system level. In the semantic structure developed in this thesis, three views on the behaviour of (components in) a compositional system are distinguished, which correspond to three levels of locality. As these views can be applied to any component at any level in a compositional system, many different levels of locality can be distinguished.
- The semantics of OSL is defined in terms of global states and implies that a notion of global time is available, which is not the case for the semantic structure developed in this thesis.

Interaction is modelled in OSL in a way that resembles interaction in the semantic structure developed in this thesis. In OSL, interaction is modelled by *event calling* as explained above. Event calling is a form of implication: the occurrence of an event described by φ in the event calling formula $\varphi \Rightarrow \psi$ implies the occurrence of an event described by ψ in another object. This is similar to the interpretation of information link mappings as described in Chapter 6: the occurrence of specific states in a local component trace of the domain of a link implies the occurrence of specific states in a local component trace of the co-domain of the link.

12.3 Local Model Semantics for Contextual Reasoning

An important topic in the study of common-sense reasoning in Artificial Intelligence is the formalisation of *contextual reasoning*. In this section, a formalisation of contextual reasoning developed by Giunchiglia and Ghidini (1998) is compared to the semantic structure developed in this thesis. Contextual reasoning is related to multi-agent systems as follows. An agent's observations are often constrained, which precludes the agent from obtaining a complete model of its environment. An agent may be aware of the fact that it (temporarily) cannot obtain observations of a specific aspect of its environment. In this case, the absence of information on this aspect can be modelled using epistemic logic or three-valued logic. However, an agent may not even be aware of the *existence* of specific aspects of its environment. Its view of its environment is local in the sense that its view cannot accommodate knowledge on the aspects of which it is not aware. The agent reasons about its environment from the context consisting of (partial or complete) knowledge of those aspects of which it is aware. This is called the *principle of locality* in (Giunchiglia & Ghidini, 1998).

In their formalisation, Giunchiglia and Ghidini (1998) only consider objective information about the environment available to agents in their context (subjective interpretations are not considered). As several agents reason about the same environment, parts of the different agents' objective information overlap. The overlapping parts of different contexts must match, as they are local views of the same part of the environment. This is called the *principle of compatibility* in (Giunchiglia & Ghidini, 1998).

Giunchiglia and Ghidini (1998) present a semantics that can be used to capture contextual reasoning according to the principles of locality and compatibility. Their approach is related to the semantic structure presented in this thesis, as locality and compatibility are the basic principles employed in both approaches.

As a starting point, Giunchiglia and Ghidini (1998) assume that local reasoning within a context is described by (first-order) local languages, a (possibly different) language L_i for each agent i. Similar to the languages presented in Chapter 9, each language only contains terms and predicates to describe the aspects that are known to exist in the local view. Associated with each local language L_i is a class of standard first-order interpretation structures $\underline{M}_i$. The formulae that describe the knowledge available in a specific local view i denote a set of local models $M_i \subseteq \underline{M}_i$ that classically satisfy those formulae.

Giunchiglia and Ghidini (1998) introduce (non-temporal) compatibility relations as relations on the sets of local models. Similar to (Giunchiglia & Ghidini, 1998), in this description of local model semantics, all definitions are presented for two agents. (It is straightforward to extend the definitions to the general case for more than two agents.) A compatibility relation is defined as a relation $\boldsymbol{C} \subseteq \mathcal{P}(\underline{M}_1) \times \mathcal{P}(\underline{M}_2)$. ($\mathcal{P}(S)$ denotes the power set of S). For convenience, Giunchiglia and Ghidini use the notation $\boldsymbol{c}_i$ for the i-th element of a tuple $\boldsymbol{c} \in \boldsymbol{C}$. A compatibility relation $\boldsymbol{C}$ is called a *model* for $\{L_1,L_2\}$ if $\boldsymbol{C} \neq \emptyset$ and $\langle \emptyset;\emptyset \rangle \notin \boldsymbol{C}$. A *context* C_i is defined as the set of local models in the compatibility relation, or formally, the set of local models $m \in \underline{M}_i$ such that $m \in \boldsymbol{c}_i$ for $\boldsymbol{c} \in \boldsymbol{C}$. In other words, a context is the set of local models that take overlap with other context as modelled by a compatibility relation, into account.

The framework described in (Giunchiglia & Gidini, 1998) is very general. Compatibility relations are not further defined as specific (proper) subsets of $\mathcal{P}(\underline{M}_1)\times\mathcal{P}(\underline{M}_2)$ (e.g., for specific applications). The role of compatibility relations is to model constraints on local models imposed by non-local dependencies. The constraints modelled by compatibility have, in the words of Giunchiglia and Ghidini (1998), “the structural effect of changing the set of local models defining each context. It forces local models to agree up to a certain extent”. This structural effect is made explicit in the notion of satisfiability of a (local) formula:

Definition 12.2. (Satisfaction (Giunchiglia & Gidini, 1998), slightly different notation). *Let* $\mathbf{C}$ *be a model and let* $\varphi\in L_i$ *be a formula. Then* $\mathbf{C}$ satisfies φ *iff for all* $\mathbf{c}\in\mathbf{C}$, $\mathbf{c}_i$ *satisfies* φ, *where* $\mathbf{c}_i$ *satisfies* φ *iff for all* $m\in\mathbf{c}_i$, *m classically satisfies* φ.

Thus, local models *m* that do not occur in pairs in the compatibility relation $\mathbf{C}$ cannot satisfy φ.

It is interesting to conclude the description of local model semantics with Giunchiglia and Ghidini’s notion of logical consequence in contextual reasoning, which is the foundation for reasoning about agents that maintain contextual views of their environment:

Definition 12.3. (Logical consequence (Giunchiglia & Gidini, 1998), slightly different notation). *Let* Γ_1 *and* Γ_2 *be sets of formulae of* L_1 *and* L_2, *respectively. A formula* $\varphi\in L_i$ *is a* logical consequence *of a set of formulae* $\Gamma=\Gamma_1\cup\Gamma_2$ *iff for all models* $\mathbf{C}$ *and for all* $\mathbf{c}\in\mathbf{C}$, *if for all* $j\in\{1,2\}$, $j\neq i$, $\mathbf{c}_j$ *satisfies* Γ_j, *then for all* $m\in\mathbf{c}_i$, *if m classically satisfies* Γ_i, *then m classically satisfies* φ.

The basic principles locality and compatibility in the semantic structure developed in this thesis are very similar to the notions of locality and compatibility employed by Giunchiglia and Ghidini. The examples and discussions of compatibility relations in (Giunchiglia & Gidini, 1998) indicate that compatibility relations, in their approach, are constructed in the same way as compatibility relations and constructs in the semantic structure presented in this thesis. Moreover, compatibility relations have exactly the same role: to model constraints on local models (local component and link traces in this thesis) imposed by non-local dependencies (information transmission in this thesis). However, a number of differences between local model semantics and the semantic structure developed in this thesis can be identified:

- The approach presented in (Giunchiglia & Ghidini, 1998) only covers static aspects of contextual reasoning. However, in this thesis, the principles of locality and compatibility relations are extended for dynamic domains and applied to the dynamics of multi-agent systems.

- As the definition of compatibility presented above indicates, Giunchiglia and Ghidini relate *sets* of local models in compatibility relations. (A compatibility relation is a subset of $\mathcal{P}(\underline{M}_1)\times\mathcal{P}(\underline{M}_2)$, not of $\underline{M}_1\times\underline{M}_2$.) Contexts consist of sets of local models, which enables the representation of partial information. As Definition 5.18 indicates, a compatibility relation consists of triples of single local component or link traces. Partiality may be modelled by using partial states inside local component and link traces (as in the DESIRE modelling framework presented in Chapter 9);
- Compatibility in local model semantics model constraints imposed by overlapping parts of local models. In the semantic structure developed in this thesis, local component and link traces do not overlap. A local component or link trace only consists of states of a single component or link. However, information transmission introduces similar constraints between local models as the overlap of local models in contextual reasoning, giving rise to a notion of (temporal) compatibility;

12.4 Other Approaches

The semantic structure developed in this thesis can be compared with many more approaches in various areas of research. A number of suggestions for further comparison are presented:

- Within the area of multi-agent systems, there is considerable attention for the specification of multi-agent system dynamics. A number of approaches are of particular importance to consider for a comparison with the semantic structure developed in this thesis. Kiss (1996) presents a very general overview of agent dynamics and its relations with traditional Distributed Artificial Intelligence topics such as planning and rationality. A semantics for an abstract agent programming language (for single agents) is presented in (Hindriks, Boer, Hoek & Meyer, 1998). In (Eijk, Boer, Hoek & Meyer, 1998), information exchange in a multi-agent system is formalised. Situation calculus (McCarthy & Hayes, 1969) is used for the specification of (multi-)agent systems using the Congolog framework in (Lespérance, Levesque, Lin, Marcu, Reiter & Scherl, 1996; Lespérance, Levesque & Ruman, 1997). Burkhard (1993) studies liveliness and safety properties in multi-agent systems. Burkhard’s (1993) approach is entirely event-based: the behaviour of a single agent is modelled by a set of strings of events, which constitutes a formal language. Liveliness and safety properties for a multi-agent system are analysed in terms of operations on formal languages.
- Chapter 7 presented a detailed comparison with the representation of concurrency in distributed systems based on the notion of dependence (Lamport, 1978). Candidates for further comparison are more algebraic approaches such as event structures (Winskel, 1989) or Chu spaces

(Pratt, 1995), or Petri net based approaches such as (Moldt & Weinberg, 1997). Ladkin and Leue (1995) and Damm and Harel (1999) study the relation between event-based and state-based descriptions of dynamics in the context of timing diagrams or message sequence diagrams. As Ladkin and Leue observe, diagrams similar to Figure 2.9 and Figure 7.4 play an important role in various research areas, including Hardware Design, Computer Networks (Tanenbaum, 1996) and Object Orientation (e.g., message sequence charts in the UML (Booch, Rumbaugh & Jacobson, 1998)).

- Semantic structures for co-ordination languages (Papadopoulos & Arbab, 1998; Ciancarini & Wolf, 1999) form another category of interesting approaches for comparison with the semantic structure developed in this thesis. As stated in Chapter 2, most co-ordination languages completely abstract from computations in a compositional system by only distinguishing a number of components, each of which contains a number of computational processes. Co-ordination languages focus on the specification of systems of such components and information exchange between them, which is similar to the semantic structure developed in this thesis. The semantical description of Manifold (Bonsangue, Arbab, Bakker, Rutten, Scutellà & Zavattaro, 1998) is of particular interest as it has a number of characteristics in common with the semantic structure developed in this thesis. In Manifold, communication channels between components are explicitly modelled. Moreover, these channels have their own state (similar to commitment presented in Section 2.2.3) and are autonomous.

12.5 Further Research

The semantic structure developed in this thesis provides a ground for further research in a number of areas. First, the semantic structure can be extended, i.e. by including additional facilities for the specification of real-time behaviour (see also Section 12.5.1), process creation, or other phenomena that have been studied in for instance the context of Process Algebra (Bergsta & Klop, 1985). Second, verification and validation of multi-agent systems represented as compositional systems using the semantic structure can be investigated. Formal verification requires a formal proof system. Section 12.5.2 sketches how two existing logics can possibly be used for reasoning about properties of multi-agent systems in the context of the semantic structure. As an aside, validation and verification of properties of multi-agent systems do not require the use of mechanical proof creation or checking. In (Jonker, Treur & Vries, 1998), an approach is presented for the verification of multi-agent systems using rigorous mathematical reasoning. As the semantic structure developed in this thesis is itself defined in mathematical terms, the approach described in (Jonker, Treur & Vries, 1998) may be particularly suitable. However, this form of verification is difficult to support directly by mechanical means, such as for instance a proof checker for general first-order logic. Third, some common topics in concurrency theory, such as notions of equivalence of behaviour, can be investigated in the context of the semantic structure. In Section 12.5.3, some directions for such investigations are briefly sketched. Fourth, additional applications of the semantic structure can be investigated. In Section 12.5.4, further research with respect to the DESIRE modelling framework is sketched.

12.5.1 Real-time Logics and Fictitious Clocks

Standard temporal logic can be used to express qualitative statements about time, but not to express quantitative statements. Real-time logics are extensions of temporal logic that support quantitative statements about time. Most real-time logics are either based on dense time (as presented in Section 12.1.3) or on fictitious clocks (Raskin & Schobbens, 1997). In the fictitious clocks approach, a global fictitious clock is introduced that is assumed to generate clock ticks at a fixed rate. The clock ticks are represented in the sequence of discrete, global states of a system. Time is measured by counting the number of states between two states in which a clock tick occurs. When dense-time temporal logic is used as a real-time logic, a metric on the dense set of time points (e.g., the standard metric on the set of real numbers) is used for quantitative statements about time.

In (Raskin & Schobbens, 1997), the relation between the dense time and fictitious clocks approaches to real-time logics is investigated. Raskin and Schobbens (1997) present one temporal logic language with two different interpretations (dense time and fictitious clocks). The most imporant connective in the logic is a special form of until, which is parameterised by an amount of time. For instance, $\phi\mathcal{U}_{\leq 3}\varphi$ is true at a time point t iff ϕ is true at t and stays true for at most three units of time, after which φ is true.

The fictitious clock interpretation of their language is based on models consisting of a discrete sequence of global states. A global state itself is a set of proposition symbols that are true in that state. In a subset of these states, the proposition symbol tick is true, which indicates that a tick of the fictitious clock occurs between that state and the next one. A formula $\phi\mathcal{U}_{\leq 3}\varphi$ is true at state s in such a model iff ϕ is true in state s and in the sequence of states between s and the state in which ϕ is no longer true, but φ is, at most three states in which ‘tick’ is true appear.

Raskin and Schobbens do not introduce a formal framework with a specific computational model, such as DESIRE or Concurrent MetateM. Instead, they only introduce the temporal logic mentioned above. However, their work is intended for the specification of properties of computer systems (as stated on page 166 of their paper). The dense time interpretation of the language is therefore based on models that consist of consecutive intervals of real time in which the (global) state of a system remains constant. The union of these intervals is required to cover an unbounded amount of time, which ensures that there cannot be an infinite number of intervals in which the state is distinct in a finite amount of time (non-Zenoness).

The semantic structures and logic language presented in (Raskin & Schobbens, 1997) are standard. The main result of the paper is a precise definition of a relationship between dense time and fictitious clock semantics. Using this relation, approximate answers to satisfiability questions for the dense time semantics (which is undecidable) can be found: dense time formulae are abstracted to fictitious clock interpretation, which is decidable. This approach can be employed at the local level in the semantic structure to mechanically determine whether specific local component or link traces satisfy a local formula.

Many differences can be distinguished between the approach by Raspin and Schobbens, and the semantic structure developed in this thesis. On the one hand, both the dense time semantics and in the fictitious clocks semantics assume that global time exists. Moreover, the approach by Raskin and Schobbens is based on a notion of global state. Locality, compositionality and hierarchical composition are not treated at all. On the other hand, the approach by Raskin and Schobbens provides much more detail with respect to the representation of real-time systems than the semantic structure presented in this thesis. Also the DESIRE modelling framework presented in Chapter 9 has no built-in facilities for expressing real-time requirements.

A possible further research question is: can the approach of Raspin and Schobbens be used to specify sets of local behaviour of individual components, and if so, is this beneficial? The definitions of local component and link traces presented in Chapter 5 enable the use of dense time or fictitious clock real time logic: for the time frames on which these traces are based, dense time or fictitious clock structures as described in (Raskin & Schobbens, 1997) can be used. A logic language for real time, such as the logic presented in (Raskin & Schobbens, 1997), could be used to specify the sets of local component and link traces.

12.5.2 Verification

The formally defined semantic structure presented in this thesis enables precise, mathematical proofs of properties of multi-agent systems. It is often desirable to mechanically verify or generate such proofs, which requires a formal logic specifically suited for the semantic structure developed in this thesis. The application of the semantic structure in the context of DESIRE presented in Chapter 9 shows how temporal logic can be used for the specification of local behaviour. The application of formal logic for reasoning at a more global level is an area for further research. Two approaches that seem to be particularly suitable for this purpose are Multi-Context Systems (Giunchiglia, 1993; Giunchiglia & Serafini, 1994) and Interleaving Set Temporal Logic (ISTL; Katz & Peled, 1990), which are both described below.

12.5.2.1 Multi-Context Systems

A Multi-Context System (Giunchiglia, 1993; Giunchiglia & Serafini, 1994), also called a Multi-Language System, is a set of (possibly different) logic languages, together with bridge rules which relate formulae in these languages. The use of different logic languages is the same as in Local Model Semantics (discussed in Section 12.3): each language is a local language that describes a specific context. A bridge rule is an inference rule in which the language of the condition of the rule is not the same as the language of the conclusion of the rule. A bridge rule states that if a formula is proven that matches the condition of the bridge rule in a derivation in one context, then the conclusion of the rule can be introduced as an assumption in a derivation in another context. Bridge rules are not the same for all Multi-Context System applications; instead they differ from system to system, depending on the application.

The Multi-Context Systems approach is a likely candidate for a formal logic that enables reasoning at the global level in applications of the semantic structure for the following reasons:

- Similar to the semantic structure developed in this thesis, the Multi-Context Systems approach focuses on locality and compositionality. In a Multi-Context System, only local languages are defined, and only local proofs can be derived. Proofs of global properties are composed of these local proofs;
- Multi-Context Systems may be considered the syntactical counterpart of Local Model Semantics (Giunchiglia & Gidini, 1998) as presented in Section 12.3, although Giunchiglia and Gidini (1998) do not investigate the relationship between Multi-Context Systems and Local Model Semantics.

The Multi-Context Systems approach is supported by the proof checker GETFOL (Giunchiglia, 1994,1993). The availability of a proof checker contributes greatly to the attractiveness of the Multi-Context Systems approach for the semantic structure developed in this thesis. However, as yet there are no research results with respect to the application of the Multi-Context Systems approach for the semantic structure. In this section, only a sketch is provided of how an application of the Multi-Context Systems approach for proving properties of multi-agent systems could be envisioned.

Figure 12.4 shows a derivation of the property $\boldsymbol{start} \rightarrow \boldsymbol{F}\mathrm{solved1}$ for the distributed problem solving system used in the discussion of Concurrent MetateM in the first section of this chapter. (In (Fisher & Wooldridge, 1997), the same property is proven using a Hilbert-style temporal logic.) Please take note that the derivation shown in Figure 12.4 is tentative and most probably unsound. The derivation is only presented to sketch the flavour of an application of the Multi-Context Systems approach. The notation used in Figure 12.4 is the same as in (Giunchiglia, 1993).

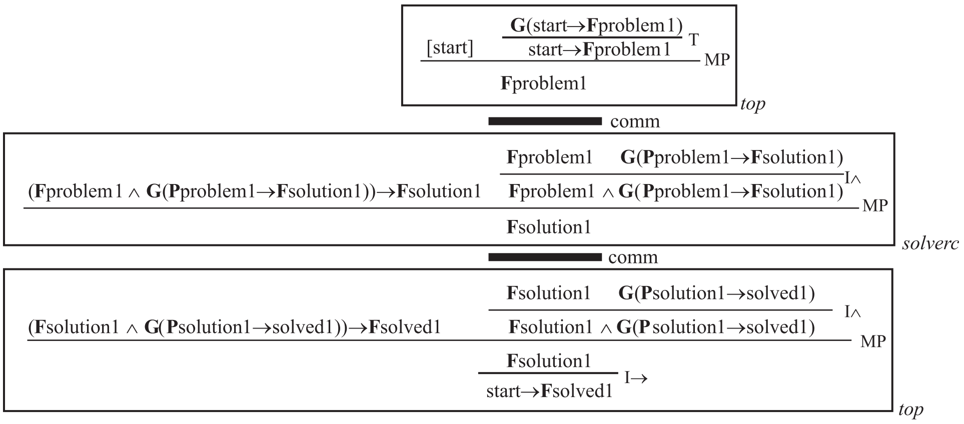

Figure 12.4: Tentative sketch of a derivation of ***start→F**solved1.*

In the derivation shown in Figure 12.4, each box corresponds with a component: both boxes labelled *top* correspond to component *top*, and the box labelled *solverc* corresponds to component *solverc*. Each box contains a sequent calculus deduction using only propositional symbols defined for the corresponding component. In other words, each box contains a derivation in the context of its corresponding component, using the local language of that component. In the derivations, three kinds of open assumptions can be distinguished:

- Assumptions of the form $\boldsymbol{G(\rho)}$, where ρ is a rule occuring in the specification of the corresponding component. (See Figure 12.1 for the specification of components *top* and *solverc*.) These assumptions can be considered to be axioms which state that in the component, all rules by which it is specified are always applicable;
- Assumptions of the form $(\boldsymbol{F}\varphi \wedge \boldsymbol{G}(\boldsymbol{P}\varphi\rightarrow\psi))\rightarrow\boldsymbol{F}\psi$ and $(\boldsymbol{F}\varphi \wedge \boldsymbol{G}(\boldsymbol{P}\varphi\rightarrow\boldsymbol{F}\psi))\rightarrow\boldsymbol{F}\psi$, where $\boldsymbol{P}\varphi\rightarrow\psi$ is a rule in the specification of the corresponding component. Assumptions of this form, in general, express *local* axioms of computation such as, e.g., liveliness. For instance, an axiom of the form $(\boldsymbol{F}\varphi \wedge \boldsymbol{G}(\boldsymbol{P}\varphi\rightarrow\psi))\rightarrow\boldsymbol{F}\psi$, where $\boldsymbol{P}\varphi\rightarrow\psi$ is a rule, states that if eventually the condition of the rule is true, then the conclusion will also eventually be true. Other assumptions in this category may for instance be frame axioms, or, in the context of Concurrent MetateM's TBL (see Section 12.1.3), axioms of the underlying epistemic logic. Further research is needed to identify which axioms are relevant in the context of multi-agent systems;
- Assumptions introduced by bridge rules, such as for example ***F**problem1* in the box labelled *solverc*. In the derivation shown in Figure 12.4, only one bridge rule is applied, called *comm* and denoted by a heavy, solid line in the figure. Bridge rules differ for specific applications of the multi-context systems approach. For the semantic structure developed in this thesis, a

> possible bridge rule may state that if an information link exists from a component *C* to a component *D* that links φ to ψ, then if $\boldsymbol{F}\varphi$ is derived in component *C*, $\boldsymbol{F}\psi$ can be introduced as an assumption in a derivation in the context of component *D*. Bridge rules in this application of the Multi-Context Systems approach are closely related to properties of information transmission as presented in Chapter 6, and to information link mapping descriptions, which specify how states in various components are related.

The tentative derivation presented in Figure 12.4 suggest an approach for proving properties of a multi-agent system using temporal local model semantics. The basic intuition that forms the basis of this approach is as follows. As the notion of compatibility relations does not enforce any relation on the local clocks of components *C* and *D*, proofs have to be based on local properties of components that state that relevant events *eventually* happen. Bridge rules ensure that these relevant events *actually* happen if, as proven for another component, a related event happens in that component. It remains to be investigated whether this is sound and sufficient for proving all valid properties.

12.5.2.2 ISTL—Interleaving Set Temporal Logic

Another likely candidate for a formal logic that enables proving properties of compositional systems is Interleaving Set Temporal Logic (ISTL; Katz & Peled, 1990). Syntactically, ISTL is similar to branching-time temporal logics such as CTL* (Emerson & Halpern, 1986). However, the interpretation of ISTL differs from most other temporal logics: in ISTL, the two sources of non-determinism found in concurrent systems are not identified, as explained below.

In a concurrent system in which no global clock is assumed, two sources of non-determinism can be distinguished. First, specific processes in a concurrent system may be non-deterministic with respect to their reactions to information received from other processes. Although it is not known beforehand which reaction will be chosen by the process, all observers of the process observe the same reaction after a choice is made. Second, due to the absence of global time, different observers observe different behaviour of the same process even if this process always reacts in the same way. In most temporal logics, both forms of non-determinism are represented in exactly the same way: as branching points in their interpretation structures. As a consequence, the two forms of non-determinism cannot be distinguished in these logics. Moreover, in these logics, the behaviour of one concurrent system is described by one branching-time interpretation structure.

In ISTL, both forms of non-determinism are distinguished. The behaviour of one concurrent system is described in ISTL by a *set* of branching-time interpretation structures. Each different branching-time interpretation structure in this set represents a different reaction to information received by a process in the system. Each branching point in one specific branching-time interpretation structure represents non-determinisn with respect to observations due to the absence of global time.

Interpretation structures of ISTL are similar to the global view on the behaviour of a compositional system presented in Chapter 7. As explained in Chapter 7, global states for one specific multitrace are partially ordered. The partial order of global states represents non-determinism of observations caused by the absence of global time. If a component in a compositional system is non-deterministic with respect to its reactions to information received from other components, the different behavious are represented by different multitraces, which leads to different sets of partially ordered global states.

It may be possible to prove that the class of interpretation structures formally defined in (Katz & Peled, 1990) for ISTL is exactly the same as the class of partially ordered sets of strict global states defined in Chapter 7. If this is the case, than ISTL is suitable as a logic to reason about global properties of a compositional system. If this is not the case, further research may try to adapt ISTL's inference rules to the semantic structure developed in this thesis.

A specific question that has to be answered in the development of formal logic systems for the semantic structure is the question of fairness (Francez, 1986; Manna & Pnueli, 1992). In fact, it may come as a surprise that fairness is not discussed in this thesis, as specific semantic structures often commit to a specific notion of fairness. However, in this thesis, the view of OSL is adopted, which states that fairness is a property of an *application* of the semantic structure and may differ for different applications. In the words of the developers of OSL: “The question of fairness was completely disregarded in this paper. Indeed, we assume that the task of imposing fairness or justice requirements is left to the specifier (to the point of inconsistency if by mistake too much is required).” (Sernadas, Sernadas & Costa, 1995, p. 627).

12.5.3 Concurrency Theory

As stated in Section 1.4.5, this thesis does not make any claim with respect to the applicability of the semantic structure as a general theory of concurrency. However, the question whether the semantic structure is applicable outside the area of multi-agent systems may be interesting for further research. A general topic in Concurrency Theory is equivalence of computations. For instance, in Process Algebra (Bergstra & Klop, 1985) approaches, a process and its desired properties are both specified using a process algebra. The question whether a property holds for the process is reduced to the question whether the computations denoted by the two specifications are equivalent. The development of equivalence notions for process algebra specifications is therefore an important research area. It would be interesting to develop similar notions for the semantic structure presented in this thesis.

12.5.4 Applications of the Semantic Structure

Some directions for further research can be identified in the area of applications of the semantic structure:

- The semantic structure fully abstracts from specific agent processes such as co-operation and maintenance of mental notions such as beliefs, desires, intentions and commitments. The specification of, for instance, a BDI architecture for a single agent, or of inter-agent commitments in a multi-agent system, raises interesting questions with respect to the application of the semantic structure. In Chapter 10 and Chapter 11, applications in these areas are presented. In these applications, the behaviour of agents does not change during the lifetime of agents. Further research may investigate how the semantic structure can support adaptive behaviour and learning;
- In the context of DESIRE, at least two directions for further research can be identified. First, DESIRE has recently been extended with support for dynamic adaptation of agent models and for additional forms of information transmission such as broadcasting. Further research may extend the formal description of DESIRE presented in Chapter 9 to include these new facilities. Second, some research questions with respect to implementation generators for DESIRE can be identified. To support the design process for multi-agents systems using DESIRE, an implementation generator has been developed. Currently, the implementation generator executes automatically generated prototype implementations of multi-agent systems on a single processor using pseudo-concurrency. To do full justice to the specifications a, prototype implementation should ideally run in a distributed environment. In the near future, the implementation generators will be augmented, providing support for concurrent execution. A possible research question is: is it possible to develop or validate such an implementation generator directly from the description of the DESIRE behaviour as presented in Chapter 9?

12.6 Conclusions

The aim of the research presented in this thesis is formulated in Chapter 1 as the development of a formal, compositional, semantic structure for multi-agent systems dynamics. Section 1.4.3 lists four requirements for the semantic structure: (1) both *deliberation and interaction* in a multi-agent system should be explicitly represented, (2) the semantic structure should support the *compositionality principle*, (3) an agent's dynamics should be described in terms of its *state and state transformations*, and (4) the semantic structure should support *locality*. This final section of the thesis describes the extent to which these requirements have been fulfilled.

The research presented in this thesis is based on a central assumption, introduced in Chapter 1: multi-agent systems are represented as compositional systems. As a consequence, the main building blocks of the semantic structure are components and information links between components. Studies in the context of DESIRE have shown that multi-agent systems can successfully be represented as compositional systems (Brazier, Dunin-Keplicz, Jennings & Treur, 1997; Brazier, Eck & Treur, 1997b,2001a; see also the concluding remarks with respect to DESIRE presented in Section 9.4). Moreover, existing generic representations (models) of multi-agent systems can be (re)used for agents and components of agents, such as the generic agent model presented in (Brazier, Jonker & Treur, 2000). In Chapter 3, modelling choices with respect to how a multi-agent system can be represented as a compositional system, are discussed. Chapter 2 discussed compositional systems and presented commitments with respect to properties of compositional systems that can be described using the semantic structure developed in this thesis.

The main constructs that comprise the semantic structure are mathematically defined in Chapter 5. The static compositional structure of components is described by *structure hierarchies*. Three views on the behaviour of a compositional system (described by a structure hierarchy) are presented: the *black box view*, the *white box view*, and the *glass box view*. Each view is a set of compatible *multitraces*. A multitrace is an indexed set of *local component and link traces* of (a subset of) the components and links in the compositional system, in which a partial order on the index set represents the compositional structure of the system. A local component or link trace consists of *local component or link states*. A local state is fully determined by one single component and link (hence the name). For each component and link, a set of local component traces is assumed to be given. Such a set contains all local component and link traces that describe possible behaviour without taking constraints imposed by information transmission into account. *Compatibility relations* group triples consisting of a local link trace for an information link *I*, a local component trace of the domain of *I*, and a local trace of the co-domain of *I*. Such triples are related by compatibility if the local traces in the triple respect constraints imposed by information transmission, e.g., if a state occurs in a local component trace of the domain in which information has to be transmitted, then a related state occurs in the co-domain of the link in which the transmitted information has just been received. A *compatible multitrace* contains only local component and link traces that occur in compatibility relations. Properties of information transmission such as the lossless transmission property are captured as properties of compatibility relations. The definitions of these properties demonstrate how compatibility relations relate local component traces according to *information link mappings*.

The four requirements for the semantic structure described in Chapter 1 and mentioned at the beginning of this section are met to the following extent:

- The first requirement (deliberation and interaction) is met to the following extent. Local component traces describe processes within primitive components that use information received from other components and produce information intended to be transmitted to other components. Local link traces and compatibility relations describe the information transmission processes needed to actually transmit information. As explained in Chapter 3, an agent's deliberation is represented by internal processes inside components, while an agent's interaction with other agents and with its environment is represented by information transmission. The semantic structure thus supports both kinds of agent activities;

- With respect to the second requirement (the compositionality principle), it is important to note that the adjective 'compositional' refers to three different but related notions in the literature. These notions are supported by the semantic structure developed in this thesis in the following way:
  - First, the adjective 'compositional' simply denotes that a system consists of components. First and foremost, the semantic structure focuses on compositional systems: systems consisting of components and information links between components. Components and information links are the two most important constructs that constitute the semantic structure. In Chapter 2, compositional systems are (informally) defined, commitments with respect to properties of compositional systems adopted in this thesis are presented and alternative properties are described. The notion of a compositional system is also discussed from the perspective of areas of research such as Software Engineering and Co-ordination Languages;
  - Second, the adjective 'compositional' is used in the following way: a compositional system is a system in which the dynamics of a system composed of a number of components is defined by a composition relation. The composition relation itself defines the dynamics of a system composed of a number of components in terms of the dynamics of those components. The definitions of the three views on the behaviour of a composed component presented in Chapter 5 (the black box, glass box and white box views) clearly indicate that compositional systems in the semantic structure adopt this compositionality property: each view is built from local component and link traces or from multitraces that describe the behaviour of lower level components. Moreover, in Chapter 5, propositions are presented that specify how more global views can be composed of local views on the behaviour of a component;
  - Third, the adjective 'compositional' may apply to proofs of properties of a system that itself may or may not be compositional (in the previous meanings). In the preface to (Roever *et al.*, 1998), the term 'compositional method' refers to "Any method by which the properties of a system can be inferred from the properies of its constituents, without additional information about the internal structure of the constituents". This

  meaning of ‘compositional’ is not directly applicable to the semantic structure developed in this thesis, as the semantic structure is not equipped with a formal logic. However, the focus of the semantic structure on other notions of compositionality most probably will facilitate the development of compositional proof methods for the semantic structure. Foster (1996) connects this view on the term ‘compositional’ with the previous one as follows: “A compositional programming system is one in which properties of program components are preserved when those components are composed in parallel with other program components.” Foster’s perspective on compositionality is difficult to combine with the property of locality in a hierarchical system, as properties of lower-level components need to be expressible at higher levels for Foster’s perspective to apply.

  In the semantic structure, not only primitive components, but also composed components are associated with a notion of local behaviour. This local behaviour is, constrained by information transmission, one of the building blocks of the actual behaviour of a component. It can be used for management of its input and output state and for accumulation and generalisation of information provided by subcomponents. This local behaviour of a composed component is in general completely independent of the behaviour of its subcomponents;

- According to the third requirement described in Chapter 1, the semantic structure should be state-based. As indicated in Chapter 1, agents in a multi-agent system are often described and analysed in terms of their mental state. Therefore, the dynamics of a multi-agent system should be described first and foremost in terms of state and state transformation, as opposed to actions or events. The semantic structure as presented in Chapter 5 and Chapter 6 is entirely defined in terms of states and traces consisting of states. Nevertheless, as described in Chapter 7, (mostly global) event-based notions for the description of dynamics, such as the notions presented in (Lamport, 1986; Charron-Bost *et al.*, 1996) are applicable within the state-based semantic structure;
- The fourth requirement states that the semantic structure should support locality. Locality is at the heart of the semantic structure. The basic units for the representation of dynamics are local component and link traces, which consist of local component and link states. The black box view and white box view provide different levels of locality with respect to the description of the behaviour of composed components. Global views are defined (in the form of the glass box view defined in Chapter 5 as well as the notion of global state defined in Chapter 7). These global views are defined in terms of local views and not the other way around. Commitments presented in Chapter 2 with respect to compositional systems also ensure that

> information is localised in components. Only information in the input and output interfaces of components is visible to other components. Moreover, this information is only visible to a limited set of agents: as the restrictions on information transmission presented in Chapter 2 indicate, only links between a component and the parent components and subcomponents of the parent component are allowed. As a result, information is kept as local as possible: information can only be distributed in a compositional system via parent components that may control which information is distributed to other components.

The semantic structure as presented in Chapter 5 and Chapter 6 provides all facilities needed to represent the control phenomenon. Chapter 8 describes the phenomenon of control in multi-agent systems and compositional systems in more depth. Some constructs of the semantic structure introduced in Chapter 5 and Chapter 6, however, needed to be refined in Chapter 8 to facilitate the representation of control in a *separated*, *domain-independent* way.

Chapters 9, 10 and 11 show how the semantic structure can be applied for different purposes. In Chapter 9, the semantic structure is used to describe the dynamics of models of multi-agent systems in the DESIRE modelling framework. In Chapter 10 and Chapter 11, two example multi-agent systems modelled using DESIRE are presented. The example systems show how the dynamics of complex multi-agent systems can be described by the semantic structure and how locality and compositionality help to reduce the complexity of such systems.

As indicated in Section 12.5, there are different directions for further research. The ultimate goal of all such research should be to meet the promise of multi-agent systems sketched in Chapter 1. The research presented in this thesis shows that the semantic structure is a first step towards rigorous, formal design and analysis of multi-agent systems necessary to reach this goal.