06261 Abstracts Collection
Foundations and Practice of Programming Multi-Agent Systems
— Dagstuhl Seminar —

Mehdi Dastani[1], John-Jules Ch. Meyer[2] and Rafael H. Bordini[3]

[1] Utrecht Univ., NL
mehdi@cs.uu.nl
[2] Utrecht Univ., NL
jj@cs.uu.nl
[3] Univ. of Durham, GB
r.bordini@durham.ac.uk

Abstract. From 25.06.06 to 30.06.06, the Dagstuhl Seminar 06261 "Foundations and Practice of Programming Multi-Agent Systems" was held in the International Conference and Research Center (IBFI), Schloss Dagstuhl. During the seminar, several participants presented their current research, and ongoing work and open problems were discussed. Abstracts of the presentations given during the seminar as well as abstracts of seminar results and ideas are put together in this paper. The first section describes the seminar topics and goals in general. Links to extended abstracts or full papers are provided, if available.

Keywords. Agent-oriented programming, Agent-oriented software engineering, Multi-agent implementation languages, Agent development tools and platforms, Semantics of agent-oriented languages, Specification and verification of multi-agent systems, Theories of multi-agent programming

06261 Executive Summary – Foundations and Practice of Programming Multi-Agent Systems

The "Foundations and Practice of Programming Multi-Agent Systems" Dagstuhl Seminar aimed at bringing together researchers interested in programming languages for multi-agent systems, agent-oriented software engineering, and various related aspects such as verification, and formal semantics. We were delighted with the result of this seminar, which gave participants a clear view of the most advanced techniques being currently investigated in research on those topics throughout the world, and also a clear understanding of all the most important open problems that need to be addressed by this research community. The seminar was particularly successful in elucidating the relationship between work being done by the "programming languages for multi-agent systems" (ProMAS)

research community and the "agent-oriented software engineering" (AOSE) research community. Even though the initiative for this seminar arose from the ProMAS community, we were delighted to attract many prominent researches from the AOSE community, which allowed us to achieve the positive result on the connection of ProMAS and AOSE research.

Joint work of: Dastani, Mehdi; Meyer, John-Jules Ch.; Bordini, Rafael H.

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/846

Powerjava: Interaction and Protocols

Matteo Baldoni (University of Torino, I)

This paper begins with the comparison of the message-sending mechanism, for communication among agents, and the method-invocation mechanism, for communication among ob jects. Then, we describe an ex- tension of the method-invocation mechanism by introducing the notion of "sender" of a message, "state" of the interaction and "protocol" using the notion of "role", as it has been introduced in the powerJava exten- sion of Java. The use of roles in communication is shown by means of an example of protocol.

Keywords: Roles, Agent communication, Java extension

Joint work of: Baldoni, Matteo; Boella, Guido; van der Torre, Leon

A priori conformance verification for guaranteeing interoperability in open environments

Matteo Baldoni (University of Torino, I)

An important issue, in an open environment like the web, is to have capability of guaranteeing the interoperability of a set of services. When the interaction scheme that the services should followed is given a priori (e.g. as a choreography or as an interaction protocol), it becomes possible to verify, before the interaction takes place, if the interactive behavior of a service (e.g. a BPEL process specification) respects it. This verification is known as “conformance test". Recently some attemps have been done for defining conformance tests w.r.t. a protocol but these approaches fail in capturing the very nature of interoperability, turning out to be too restrictive. In this work we will give a simple definition of interaction protocol based on message exchange and on finite state automata, and we will focus on the attempt of capturing those properties that are essential to the verification the interoperability of a set of services. In particular, our desire is to define a conformance test that can guarantee a priori the interoperability of a set of peers by verifying properties of the single peer against the choreography.

This capability is particularly relevant in open environments, like the web, where services are identified and composed on demand and dynamically and the system as a whole cannot be analysed.

Keywords: Web service interaction protocols, conformance test, interoperability, formal verification, finite state automata

Joint work of: Baldoni, Matteo; Baroglio, Cristina; Martelli, Alberto; Patti, Viviana

Model Checking Multi-Agent Programs

Rafael Bordini (University of Durham, GB)

In this talk, I shall briefly discuss our approach to model checking multi-agent software written in AgentSpeak, a logic-based agent-oriented programming language. We use existing model checkers, in particular Spin and JPF, so the approach is to translate AgentSpeak programs into Promela or Java, respectively. The main part of the talk will present in detail a property-based slicing algorithm which can significantly reduce the state space of systems to be verified by model checking. I shall conclude the talk by discussing the planned direction for this research.

Joint work with Willem Visser (NASA-Ames), Michael Fisher and Mike Wooldridge (Univ. Liverpool).

Keywords: Model Checking, Property-Based Slicing, Logic Programming, Agent Programming, AgentSpeak, Spin, JPF

Combining agents with mainstream enterprise software technologies

Lars Braubach (Universität Hamburg, D)

This talk will present work on the integration of software agents with existing mainstream software technology used in enterprise scenarios. Enterprise scenarios pose special requirements on the employed software which has to be met by agents if one wants to promote their usage in real world business applications.

Most importantly, business applications can only be built if the following non-functional requirements are addressed by the technology of choice:

- Software qualities like Scalability, transaction support, fault tolerance, reliability
- Integration with existing technology, e.g. database connectivity, web interfaces, etc.

Today, J2EE application servers represent one state-of-the-art technology to solve these problems. Nevertheless, we think that the component-oriented approach of application servers has certain drawbacks inherent to the object-oriented application conception and development, e.g. the inflexible control flow, cumbersome method-based remote interactions, etc.

If we want agents to become the successor paradigm of object-orientation we will have to show on the one hand what agents add to the portfolio of possibilities and on the other hand that agent technology is able to address the characteristics mentioned above at least as good as existing technologies.

Basically, there are two approaches how this can be achieved. One is to develop new mechanisms and software tools tailored to agents needs (e.g. the JADE approach). Another more resource sparing approach consists in adapting agents to these enterprise environments (e.g. making agents executable in J2EE servers, like the Whitestein or Agentis approach). We will outline our experiences with the integration of both technologies by two examples:

- The conceptualization of a web framework for agents.
- The conceptualization of a J2EE adapter for agents.

Joint work of: Braubach, Lars; Lamersdorf, Winfried; Pukahr, Alexander

Choosing a Suitable Logical Formalism for Verifying Multi-Agent Programs

Jan M. Broersen (Utrecht University, NL)

The aim of agent programming languages such as 3APL is to facilitate the use of cognitive concepts like 'belief', 'goal' and 'plan' for programming multi-agent systems. In this talk we investigate the requirements for choosing a suitable logical formalism for verifying such programs. We address several basic questions. For instance, how do we account for agency? Should we use a so called 'STIT'-logic? Should we use an endogenous or an exogenous temporal formalism? And if we use an exogenous language, how do we represent actions? Should we use FOL? Can we use PDL or ATL? A general answer to all these questions is that it depends on the level of abstraction we want to choose, leaving us with the question: what is a suitable level of abstraction for a cognitive agent property language? We will settle on an extension of ATL with STIT operators for strategic action.

Keywords: Agents, verification, ATL, stit

2APL: A Practical Agent Programming Language

Mehdi Dastani (Utrecht University, NL)

In this talk, an agent-oriented programming language, called 2APL (A Practical Agent Programming Language), will be introduced. This programming language facilitates the implementation of multi-agent systems consisting of individual cognitive agents.

2APL distinguishes itself from other agent-oriented programming languages by introducing both declarative goals and events (which are used interchangeably in other programming languages), and by providing practical programming constructs. These programming constructs are designed based on our experience with the existing agent-oriented programming languages used by the students in multi-agent courses. Examples are programming constructs that implement beliefs, goals, plans, events, messages, plan generation, recursion, execution of plans with critical-regions (executing plans avoiding to interleave its actions with the actions of other plans), plan revision when they fail to execute, processing internal and external events, and (adopting and dropping) goal operations. The interpreter of 2APL is built on the JADE platform which allows us to use many tools provided by the JADE.

Producing Multi-Agent Systems from a Model Driven Perspective

Jorge J. Gomez-Sanz (Univ. Comp. de Madrid, E)

Model driven development deals with the automated or semi-automated production of software systems starting with an initial specification in form of a data model. As original contribution, this development approach provides technological and theoretical support for the transition from the specification to implementation. Such results, adapted to our domain, would bring enormous benefits. Trying to clarify the benefits and drawbacks of model driven approaches, the talk will review possible applications of this paradigm to Multi-Agent systems and how this change of perspective may influence our current view of Multi-Agent Systems implementation.

Model Driven Development and AOSE

Jorge J. Gomez-Sanz (Univ. Comp. de Madrid, E)

A description of the relationships between AOSE and Model Driven Development approach, exemplified by some concrete results of AOSE. At the end, INGENIAS methodology is presented as a Model Driven solution, to show how MDD can explain what is done inside an AOSE methodology.

Keywords: Model Driven Development, Agent Oriented Software Engineering

GOAL-Oriented Agent Language

Koen V. Hindriks (Radboud University of Nijmegen, NL)

The Goal-Oriented Agent Language (GOAL) is a programming language that implements the core of the agent-oriented programming paradigm.

The programming language includes constructs for beliefs, goals and actions. A commitment strategy wrt goals is built into the semantics to ensure stability of the goals an agent pursues and to ensure an agent drops its goals when they have been achieved. GOAL programs can be verified using a corresponding formal agent logic.

Agents and Services — Commonalities and Differences

Benjamin Hirsch (TU Berlin, D)

This talk covers two subjects. Firstly, a broad overview over service oriented architecture (SOA) is given, and the common theme between SOA and agents is extracted. Aim of this is to encourage the cross-fertilisation of the two different fields.

The second part of the talk introduces JIAC IV, an agent based serviceware framework, which has been developed at TU Berlin.

Keywords: JIAC, Agents, SOA

Programming MAS reorganisation with MOISE+

Jomi Fred Hübner (Universidade Regional - Blumenau, BR)

The MOISE+ model allow us to explicitly represent the organisation of an MAS based on three dimensions: structural (groups, roles, relationships), functional (global plans, goals, missions), and deontic (obligations, permissions). Its main feature is the independence between the structural and the deontic dimensions, it thus simplifies changes in these two specifications. Using the MOISE+, we are also able to define an organisational structure bearing on a reorganisation process along four phases: monitoring (when to reorganise), design (ways of building a new organisation), selection (how to choose an organisation), and implementation (how to change the current running organisation).

Keywords: Multiagent Systems, Organisation of MAS, Reorganisation of MAS

Easy yet Hard. Model Checking Abilities of Agents: Complexity Results

Wojtek Jamroga (TU Clausthal, D)

Alternating-time Temporal Logic (ATL) is a logic that allows to express facts about strategic abilities of agents and their teams in multi-agent systems. Model checking ATL formulae, if feasible, can be an important tool for verification of MAS, but it can be also used e.g. for planning in multi-agent domains.

A nice (and widely appreciated) result says that model checking ATL is tractable if the model is given explicitly. That is, the problem is linear in the number of transitions in the model, and the length of the formula. However, a more careful anaysis reveals that model checking ATL is not polynomial any more when we take the number of states (and agents) as input parameters rather than the number of transitions. Thus, a seemingly slight change of the setting renders the problem intractable. Moreover, model checking ATL is EXPTIME-complete for a higer-level description of multi-agent systems (so called concurrent programs). This can be seen as a note of warning for those who tend to take complexity results without any reservations. Sure, they are important, but one must remember that they are always relative to the context in which the problem is defined.

Agents as realisers of organisations

Catholijn Jonker (Radboud University of Nijmegen, NL)

In this talk I will will motivate my extension of research into agent design to organisation design. The increasing complexity of the multi-agent systems being developed, drives the need for more abstraction. Furthermore, in the multi-agent system design, typically, the human players are not considered explicitly in the design.

By abstracting from the agents, and first considering the organisational design (with groups, roles, and interactions between them), the organisation of the multi-agent system becomes more clear, and the requirements that have to be met by agents (human, software or otherwise) can be presented per role. Furthermore, the number of instantions needed per role can be considered without thinking yet of which agent will play that role or a specific role instantiation. Simulation at the level of the organisation without bothering about the agents yet, provides additional insight in the correctness/flaws. In my proposal, the agents are the realisers of organisations. So, if the organisational design is validated, then the agents can be assigned to the roles, or, designed to fit the role properties. I will give some examples, to clarify my points.

A second point made in my talk is to not forget the humans involved in the modelling process. Organisation designers, stakeholders, domain experts, hci designers, all have their own concerns and views on the system and the modelling thereof. As a result the requirements they have on the presentation of the model and the key elements therein differ. Furthermore, in the presentation also the cognitive capabilities of humans to absorb information have to be taken into account.

A third point is that organisations are dynamic in nature. They adapt to a changing environment, the agents playing a role in the organisation can learn, and on the basis of that also the role behaviour might change over time. The same holds for gorup and organisation behaviour. Roles and groups can be added

or disappear and the agent allocation has to be adapted accordingly. Furthermore, agents might be damaged or killed making the role realisation faulty or non-existent. In such cases what is the status of the system, what of the role? Finally, the dynamics of organisations make presenting the organisations with their dynamics a real challenge.

Keywords: Organisation modelling, organisational awareness, organisation dynamics, human perspective

See also: Hoogendoorn, M., Jonker, C.M., Schut, M.C., and Treur, J., Modeling Centralized Organisation of Organizational Change. Jounal of Computational and Mathematical Organisation Theory. In press, 2006.

Representing goals with Answer-set Programming

João A. Leite (Universidade Nova de Lisboa, P)

In this talk we will explore the use of Answer-set Programming, and several of its extensions to deal with knowledge evolution, to represent and reason about the goals of an agent.

Symbolic Verification of Multi-Agent Systems

Alessio R. Lomuscio (University College London, GB)

Multi-agent systems are open, highly-autonomous systems whose components act rationally, independently or interacting with their peers, to achieve their design objectives. While several formalisms in Artificial Intelligence have been developed in the past to represent multi-agent systems, the issue of their automatic verification has acquired prominence only very recently. In this talk I will try to discuss some of our own contribution to this area. In particular I will survey some recent work on a family of temporal, epistemic, correctness, ATL logics as well as symbolic model checking techniques (both obdd- and sat-based) for their verification. I will discuss current research directions and, if time permits, present a brief demonstration of MCMAS a symbolic model checker for Multi-Agent systems released under GPL.

Joint work of: Lomuscio, Alessio R.; Penczek, W.: Raimondi, F.; Wozna, B.

A Tool for Engineering and Implementing Agents that Follow Agent-Interaction Protocols Specified in WS-BPEL

Viviana Mascardi (University of Genova, I)

The Web Services (WS) technology is currently gaining a wider and wider consensus. The features that characterise WSs, namely heterogeneity, distribution, openness, highly dynamic interactions, are some among the key characteristics of intelligent agents and MASs.

In our talk, we suggest that agents may provide both the coordination framework and the engineering metaphor to be exploited for realising complex applications lying on the WSs infrastructure. Based on our claim, we propose to use AUML to model agent interaction protocols, and standard languages for orchestrating WSs to publish the specification of these protocols on the Web. To demonstrate the feasibility of our approach, we have designed and implemented a tool that automatically creates WS-BPEL and WSDL specifications corresponding to AUML interaction protocols. Moreover our tool outputs a Prolog based representation of the protocol, that our publisher agent is able to use for generating its own protocol-compliant code.

Finally, reader agents that interact with the publisher following the protocol expressed by the WS-BPEL and WSDL specification have been implemented, and a prototype of the "publisher-reader MAS" has been built and tested using the JADE platform.

Keywords: Intelligent Agents, Web Services, AUML, WSDL, WS-BPEL, Agent-Oriented Software Engineering, JADE, Prolog

Joint work of: Mascardi, Viviana; Casella, Giovanni

Full Paper:
http://www.disi.unige.it/person/MascardiV/Software/AUML2WS-BPEL.html

Evolving and reflective expectation as a fundamental concept for the development of open multiagent systems

Matthias Nickles (TU München, D)

Traditional approaches to the engineering of multiagent systems often assume benevolence or trustworthiness of agents, or they are based on more or less rigid control mechanisms that can seriously limit the flexibility and adaptivity of the respective application. This talk presents a foundational approach to the design of open multiagent systems which fully preserves the autonomy of the agents, based on the insight that interaction structures consist of predictive or adaptive-normative behavioral expectations. Expectation graphs are introduced for the probabilistic representation of interaction structures, together with a framework for the empirical derivation, dissemination and evolution of expectations from a system designer's point of view. Besides this elementary approach, expectation-based agent holons are introduced that allow for the gradual uncoupling of agent and system behavior in order to achieve reliable system functionality without restricting the agents' autonomy.

Keywords: Multiagent systems, agent communication, interaction patterns, expectations, norms, holons

Modular agent programming language

Peter Novak (TU Clausthal, D)

After briefly introducing our modular BDI architecture, I will try to sketch a simple programming language which allows exploiting the strengths of the introduced architecture. In its core, it employs production rules of the form "if query on a mental attitude X is true, then perform an update of a mental attitude Y". I will also present our on-going work on an interpreter for such programs.

Modular BDI programming language

Peter Novak (TU Clausthal, D)

Development of practically oriented, yet semantically clearly specified agent programming languages is one of the major interests in the agent community.

In this talk, I try to analyze existing agent programming languages of AgentSpeak(L) and 3APL family. As a result of this analysis, a modular BDI architecture based on the idea of wrapping arbitrary external code in BDI components is presented. The core idea of an agent program within this architecture is a set of rules of the form "if the query ϕ to a BDI components holds, then an update on another component is executed".

A simple programming language following the design principles stated above is introduced together with a set of extensions aiming at practical usability of the language involving support for structural decomposition, deliberation cycle control and modularity.

Keywords: Agwent architectures, agent programming languages, BDI

Joint work of: Novak, Peter; Dix, Juergen

A&A: An approach for reconciling agent-orientation and object-orientation for the design and development of software systems

Alessandro Ricci (University of Bologna, I)

The work introduces the A & A (Agents & Artifacts) approach aiming at integrating at the conceptual and engineering level the agent and object-oriented paradigm in the mainstream design and programming of software systems.

On the one side, according to state of the art in agent-based computing, agent and MAS - differently from other mainstream paradigms - provide a suitable level of abstraction for designing and building complex software systems. On the other side, object-orientation still remains the reference paradigm both

in mainstream software development - as basic technology to implement service-oriented and component-oriented architectures - and in research literature concerning programming languages and software engineering, with some well-known extensions such as aspect-orientation.

However, the forecoming - partially ongoing - turn of mainstream informatics toward concurrency and distributed computing is going to change dramatically this scenario: The engineering of any kind of non-trivial applications will demand methods and technologies to systematically deal with aspects that rely and grow upon distribution of control and loosely coupled interaction, as basic assumption. Service-oriented architectures - and related standard such as WS-* - are a primary example of such a (r)evolution, and clearly would benefit by methodologies and implementing technologies that directly deal with such issues as primary aspects.

The A & A approach proposed in this work aims at providing a frame to reconcile agents and object-orientation towards the definition of a new paradigm for mainstream software development. The approach is based on the adoption of agents and artifacts as two complementary first-class abstractions, the former used to model pro-active entities encapsulating the autonoumous execution of task/goal oriented activities, the latter used to model objects and tools that are constructed, shared and (co-)used by agents in their work, constituting first-class bricks to structure and model agent computational environment.

The discussion of the the concepts will be supported by the simpA computational model and the Cartago infrastructure as basic frameworks and technologies supporting the A & A model.

Keywords: Agents Artifacts A&A

Hierarchical Planning in BDI Agent Programming Languages: A Formal Approach

Sebastian Sardina (RMIT University - Melbourne, AU)

This work provides a general mechanism and a solid theoretical basis for performing planning within Belief-Desire-Intention (BDI) agents. BDI agent systems have emerged as one of the most widely used approaches to implementing intelligent behavior in complex dynamic domains, in addition to which they have a strong theoretical background. However, these systems either do not include any built-in capacity for "lookahead" type of planning or they do it only at the implementation level without any precise defined semantics. In some situations, the ability to plan ahead is clearly desirable or even mandatory for ensuring success. Also, a precise definition of how planning can be integrated into a BDI system is highly desirable. By building on the underlying similarities between BDI systems and Hierarchical Task Network (HTN) planners, we present a formal semantics for a BDI agent programming language which cleanly incorporates

HTN-style planning as a built-in feature. We argue that the resulting integrated agent programming language combines the advantages of both BDI agent systems and hierarchical offline planners.

Joint work of: Sardina, Sebastian; de Silva, Lavindra; Padgham, Lin

Drop the Impossible Dream

Steven Shapiro (Universität Leipzig, D)

In previous work, we examined goals and goal change in the framework of Reiter's action theories in the situation calculus. One drawback of that work was that agents remained committed to their goals until they were explicitly requested to drop them. Therefore, the agents would not drop goals even if they know that they are impossible to achieve, unless explicitly requested to do so. The current framework addresses this deficiency and allows agents to drop goals thought to be impossible to achieve.

The World According To Brahms: Modeling and Simulating Work Practice

Maarten Sierhuis (NASA (RIACS) - Moffett Field, USA)

Representing how people do work can be done at many different levels. In the knowledge engineering and AI world, people's work has been described in terms of their problem-solving expertise. There, the theory is that we can model people's problem-solving behavior by representing this behavior in a computational model that is able to duplicate some of this behavior. Work process models, such as Petri-Net models of a work process, describe what tasks are performed and when. In workflow models we describe how a specific product "flows" through an organization's work process. This describes the sequential tasks in the work process that "touch" a work-product. All these modeling approaches describe the work in a system at a certain level of detail. However, what is missing from all these types of modeling approaches is a representation of how work gets done. What is missing is a description of the work at the work practice level. Work practice includes those aspects of the work process that make people behave a certain way in a specific situation, and at a specific moment in time. To describe people's situation-specific behavior we need to include those aspects of the situation that explain the influence on the activity behavior of individuals (in contrast with problem-solving behavior), such as: situated activities, agent's beliefs, world facts, tools and artifacts, geography, situational effects on the activities, communication between agents and the use of communication tools, and agent reasoning.

Brahms is a tool for modeling and simulating the way people work and collaborate, and use systems to accomplish their tasks. Brahms can be used to

describe current and future work processes and practices in human and other types of organizations. Another application of Brahms is to design the collaborative activities between multiple intelligent agentsâĂŤhuman- and software agents.

Brahms consists of a number of tools to develop simulation models of the activities and collaboration of multiple agents:

- A multi-agent modeling language for modeling agents.
- A multi-agent discrete-event simulation engine for executing (i.e. simulation or real-time execution) Brahms models.
- A relational simulation history database for capturing the data from a simulation for later analysis.
- A multi-agent time-line view of the activities of agents, including the communication interaction between agents.

With Brahms we can model and simulate how people and systems work and interact. There are many tools and languages for describing and designing the formal specification of system behavior. However, there are not many systems that are able to describe the intelligent and social behavior of humans, as well as the cooperative and collaborative behavior between humans and systems, and the situational interaction with the environment. The Brahms environment includes aspects of human and system behavior and interaction. This means that with Brahms we can model and simulate how things happen in the real world. With Brahms it is possible to use a formal modeling and simulation approach as an analysis and design methodology in the development of Cooperative Multi-Agent Systems.

In this talk, I will cover the theoretical underpinnings of Brahms, as well as explain the representational capabilities of the language, by using examples from my Human-Centered Computing research at NASA Ames Research Center.

Extended Presentation of talk

Maarten Sierhuis (NASA (RIACS) - Moffett Field, USA)

This a PDF document of my talk given at the seminar. This presentation is an extended version of the actual presentation during the seminar. It includes slides that I had to delete for the talk, due to time constraints.

For an abstract of the talk, see the abstract document on this same web-site.

Keywords: Brahms, Modeling, Simulation, Multi-Agents, Agent Oriented Programming, Work Process, Practice

Full Paper:
http://homepage.mac.com/msierhuis/Publications%202005.htm

Brahms Website: http://www.agentisolutions.com

Maarten Sierhuis (NASA (RIACS) - Moffett Field, USA)

This is the Brahms website from which you can download Brahms and get more information about the language and the tools.

http://www.agentisolutions.com

Experiences with Agent-Oriented Software Modelling

Leon Sterling (University of Melbourne, AU)

A challenge for the agent community is to make software development methodologies that are accessible to the broad community of software acquirers and developers. This talk will present some experience with developing software models using the ROADMAP methodology, more recently integrated with Prometheus. The basic high-level concepts are goals, roles and quality goals. Goal models and role models will be demonstrated through examples. The approach has been used in a variety of domains including telecommunications, casual tutor management, and the smart home.

The Role of Emotions in Agent Deliberation

Bas Steunebrink (Utrecht University, NL)

Many branches of computer science in general and multi-agent systems in particular involve problems that need heuristics in order to manage their complexities. We will discuss three problems in multi-agent systems for which we propose heuristics inspired by human emotions. These problems are related to the autonomy of individual agents, the cooperation and coordination between agents, and the believability and efficacious interaction with agents. We will show how we plan to address these problems by formalizing and implementing a model of affect based on human emotions. Specifically, we will highlight the role of emotions in the deliberation process of 3APL agents.

Goal-Oriented Modularity in Agent Programming

Birna Van Riemsdijk (Utrecht University, NL)

Modularization is widely recognized as a central issue in software engineering. We address the issue of modularization in cognitive agent programming languages. We discuss existing approaches to modularity in cognitive agent programming. Then, we propose a new kind of modularity, i.e., goal-oriented modularity, which takes the goals of an agent as the basis for modularization. Further, we present a formal semantics of goal-oriented modularity in the context of the 3APL agent programming language.

Goal-Oriented Modularity in Agent Programming

Birna Van Riemsdijk (Utrecht University, NL)

Modularization is widely recognized as a central issue in software engineering. In this paper we address the issue of modularization in cognitive agent programming languages. We discuss existing approaches to modularity in cognitive agent programming. Then, we propose a new kind of modularity, i.e., goal-oriented modularity, which takes the goals of an agent as the basis for modularization. Further, we present a formal semantics of goal-oriented modularity in the context of the 3APL agent programming language.

Keywords: Software engineering, programming language, agent, modularization, goal

Joint work of: Van Riemsdijk, Birna; Dastani, Mehdi; de Boer, Frank; Meyer, John-Jules

Integrating Methodology & Implementation

Michael Winikoff (RMIT University - Melbourne, AU)

Building an agent system begins with analysis and design, using a suitable software engineering methodology. However, ultimately the system has to be implemented using some programming language, agent-oriented or not. In this talk I discuss the "gap" between design methodologies and implementation, and present work on bridging this gap.

Integrating Methodology and Implementation

Michael Winikoff (RMIT University - Melbourne, AU)

Building an agent system begins with analysis and design, using a suitable software engineering methodology. However, ultimately the system has to be implemented using some programming language, agent-oriented or not. In this talk I discuss the "gap" between design methodologies and implementation, and present work on bridging this gap.

Survivability of multiagent systems

Yingqian Zhang (TU Delft, NL)

We studied how to optimally deploy a set of agents over a resource bounded, dynamic network such that the resulting MAS system can best survive against failures and attacks.

Based on the idea of agent replication, we built a probabilistic survivability model. We showed that the problem of finding the optimal MAS deployment is at least NP-hard. We introduced exact algorithms and fast approximations to compute the survivability of a MAS deployment and thus to guide the agent deployment. In addition, we proposed several distributed algorithms, which are able to efficiently re-deploy agents to new locations when there is a need to re-evaluate the survivability of the current MAS deployment due to the changes in the environment. We implemented algorithms, and showed experimental results.