Engineering Coordination: Selection of Coordination Mechanisms

René Schumann

National Institute of Informatics,
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
schumann@nii.ac.jp

Abstract. Reuse of code and concepts is a driving force of agent-oriented software engineering (AOSE). In the field of AOSE the probably most recognized types of reuse are agent frameworks and the FIPA standards. To support developers of multiagent systems it is also necessary to foster reuse of mechanisms like coordination. In particular we address in this article the selection of effective and efficient mechanisms for the coordination of plans among autonomous agents. We will detail research done in the field of AOSE concerning the reuse of concepts and focus on the selection of suitable mechanisms. The selection for coordination mechanisms is, up to now, not covered in AOSE sufficiently. Therefore, we present the ECo-CoPS approach that defines a structured process for the selection of coordination mechanisms for autonomous planning systems, where the local autonomy, as well as, the existing planning systems can be preserved. A case study is presented to detail how the ECo-CoPS approach can foster the selection process.

1 Motivation

Reuse of code and concepts is a driving force in software engineering in general and in AOSE in particular. In the field of AOSE the probably most recognized types of reuse are agent frameworks and the FIPA standards [7]. In this article we address the reuse of concepts on a higher level. In particular we are focusing on the reuse of coordination mechanisms for plans among autonomous agents. Each agent is representing a particular sub-plan, that needs to be coordinated with the other sub-plans. Within each plan the future actions of an agent, or the activities of the entity the agent represents, are fixed. Typically the generation of each of these sub-plans is characterized by its computational complexity and the need for information concerning goals and resources of the particular agent. The reuse of coordination mechanisms has been supported by mediating agent infrastructures like TuSCoN [16]. In these infrastructures coordination mechanisms are embedded into the environment as coordination artifacts that can be used by the agents. This is a valuable approach for reusing coordination mechanism, but some prerequisites have to be fulfilled to use these artifacts. In fact, it is necessary for the agent to reveal all planning relevant information to the infrastructure, to allow for an externalized coordination, and ensure that the

global, as well as, the local plans are feasible. This can be a serious limitation for the application of coordination artifacts. It limits the autonomy of the agents, because a significant aspect of an autonomous agent is its ability to determine its future actions, and so its local plan. Thus coordination artifacts cannot be applied to scenarios where autonomous planning agents have to coordinate their local plans to archive a joint global goal, as they have to maintain their local autonomy and they are responsible to maintain their local goals.

The field of coordination mechanisms among autonomous agents and their plans has attracted numerous researchers. Therefore, a variety of different coordination mechanisms have been proposed. Unfortunately, if a particular situation is given for which we have to select an appropriate coordination mechanism, AOSE cannot provide any guidance and therefore the decision mainly relies on the background of the developer. We have surveyed the proceedings of the previous AOSE workshops and other AOSE related literature. Moreover, we have surveyed specific reuse centered research in the field of software engineering, like the proceedings of the International Conference of Software Reuse[1]. The reuse specific research in software engineering has addressed specifically the selection of commercial off-the-shelf software. The field of AOSE has not been recoginized by those reseacher, up to now. In the field of AOSE ifself, the work published addressing the selection of existing concepts for reuse is considerable small.

In the following we will discuss work we have identified as relevant. Then we will present the ECo-CoPS approach (Section 3), which has been developed to select an effective and efficient coordination mechanism for autonomous agents, that have to coordinate their plans. The ECo-CoPS approach support developers by providing a structured decision making process and offers tooling supporting the process. In Section 4 we present an example how the process can be used. Finally we summarize our findings and outline future research.

2 Reuse of concepts in AOSE: An overview

An established way to reuse concepts in software engineering is to define and use patterns, e.g., the well-known design patterns by Gamma et al. [8]. For the field of AOSE Lind [15] suggested a format for agents oriented patterns and presents an architectural and an interaction protocol as examples. Architectural patterns, like Broker, Moderator, and Wrapper, have also been presented by Heyden et al. [10]. Interaction patterns, like patterns for Subscription or Call for proposals have been discussed by Kolp et al. [14], as well. Those authors termed these patterns *social patterns*. The authors present a framework for describing those patterns in a unified way, that has been specialized for agent-based development. Those architectural and interaction patterns have been standardized by the FIPA [7]. A wider scope of patterns in AOSE has been proposed by Sauvage [17]. Sauvage distincts between MetaPatterns, that describe abstract constructs for the design of agent-based systems. He introduces organizational schemes,

[1] For an overview of the proceedings see `http://www.isase.us/pastconferences.htm`, Accessed: 02/04/2011.

like organizations and roles, and protocols as two meta-patterns. The second group of patterns are so-called metaphoric patterns. Sauvage mentioned marks, like pheromones, and influences as two metaphoric patterns. The third class of patterns are architectural patterns, addressing the architecture of agents.

Design patterns for self-organizing systems have been summarized by Gardelli et al. [9]. The patterns are collected from the design of nature-inspired self-organizing systems. For instance, patterns addressing the evaporation, aggregation, and diffusion of pheromones are presented. The identification of particular design patterns for the coordination in self-organizing systems is addressed in the article by de Wolf and Holvoet [20]. The authors present two design patterns for coordination. These techniques are gradient fields and market-based control.

An idea for reusing proofs within the validation of multiagent systems based on the idea of the component-based verification has been proposed by Brazier et al. [2]. Thus, only the proofs for components that have changed have to be updated. If it is possible to proof that these subsystems stay within their previously defined specification, the proof of the overall systems specification remains valid. Hilarie et al. [11] argue that to facilitate reuse of agents or components of agents, it is necessary to formally specify these components and then proof the compliance of components to their specification. This can foster reuse, as those components can become the building blocks for future systems. For that reason the authors present a formal notation combined out of Objective-Z and statecharts. The focus of the authors is on the design of the formal notation and on the prove of compliance. A quite similar approach for the reuse of organizations has been proposed by Jonker et al. [12]. In their paper the authors present a formalism to describe organizational structures and they assign properties to these organizations. They propose to build a library of organizational structures. An organizational designer then should be able to place queries to the library and retrieve possible organizational structures that might suit his requirements. The authors propose different aspects for indexing, e.g., by group functionality, environment assumptions or realization constraints [12]. The retrieved organization descriptions might be adapted to the given situation at hand.

Bartolini et al. [1] argue that the current representation of interaction protocols is not sufficient, as only the sequence of messages are fixed. But typically more information is required, e.g., to generate a valid bid in an English auction. This kind of information has to be implicitly encoded by the agent designer. Thus the authors present a framework for specifying negotiations, based on rules for encoding the negotiation protocol. Agent should be able to reason about those protocols and apply them autonomously. The reuse is thereby on emphasizing a more precise and complete form of specification for negotiations.

2.1 Reuse of coordination mechanisms

An idea, already mentioned, for the reuse of coordination mechanisms are coordination artifacts [16]. Within a coordination artifact a coordination mechanism is embedded that can be used by the agents. These artifacts can be reused. As

already discussed a significant drawback of these artifacts for autonomous planning agents is that the agents must reveal planning relevant information and loose partly their autonomy about their future activities.

The need for an easier retrieval for reusing interaction protocols has been identified by Bussmann et al. [4]. Therefore they focus on the selection process of interaction protocols. To be applicable an interaction protocol has to respect the existing dependencies of the current situation. Therefore, the authors suggest to classify interaction protocols according to a number of criteria. These characteristics are: the number of agents involved, the computability of constraints and preferences, the number of agent roles, the role assignment, the number of joint commitments, and the size of joint commitment as criteria. An agent designer should specify its requirements according to these criteria and then identify an interaction protocol that might be suitable for the given situation.

As one can see in the work addressing reuse of concepts by Bussmann et al. [4] and Jonker et al. [12] the idea of building repositories that can be browsed for content with specific characteristics can be an useful approach, that has been adopted for identifiying possible suitable coordination mechanisms.

3 The ECo-CoPS approach

The main idea of the ECo-CoPS approach is, that existing planning (sub)systems should not be replaced or changed, to enable the coordination among the agents. Each agent can be the representative of a planning entity, like a company for instance. The agents can manipulate the input of the local planning system and gather information from the output of the planning system.

The goal of the ECo-CoPS approach is to guide the selection process to find a coordination mechanism for inherently distributed autonomous planning systems. This selection is guided by the ECo (**E**ngineering **Co**ordination) process that is detailed in the following. An important step of the ECo process is the prototypical implementation of possible candidate solutions. The implementation step of the ECo process is supported by the CoPS (**Co**ordination of **P**lanning **S**ystems) process and framework. Both guide and ease the implementation of a coordination mechanism and will briefly described in the following. Even though the CoPS process and the CoPS framework have been designed to support the ECo process, they are optional for the ECo process. A detailed description of the ECo-CoPS approach can be found in [18].

3.1 The ECo process

The ECo process comprises of five steps that can be executed in an iterative manner. These steps are: model the coordination problem, elicit coordination requirements, select appropriate coordination mechanisms, implement selected approaches, and evaluate candidate mechanisms to identify the best one. The process is outlined in Figure 1.

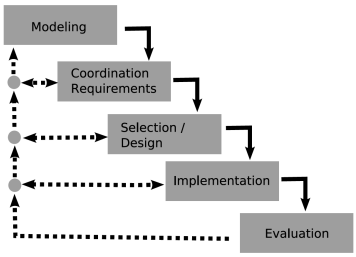


Fig. 1: The ECo process model

In the modeling phase the coordination problem and each planning problem is modeled with a specific level of detail to describe the necessary criteria that the local plans are feasible and aspects that should be optimized in the particular planning process. Moreover, the global perspective, defined by the dependencies between the planning problems, has to be modeled, as well.

In the elicitation step requirements are identified that have to be fulfilled by a coordination mechanism to be applicable for the given coordination problem. These requirements can, for instance, characterize under which conditions the planning systems are coordinated. These requirements can be formally described, using the terms and concepts introduced in the modeling step.

The third step is the selection phase. Coordination mechanisms have to be identified that can satisfy the coordination requirements. This steps results in a set of candidate mechanism that can effectively coordinate the planning systems. If this set is empty a suitable mechanism has to be designed.

To evaluate the effectiveness of these candidates they have to be implemented. Implementation is supported by the CoPS process and the CoPS framework, both discussed in the following.

If prototypical implementations exist, the candidate solutions can be evaluated with real-world like data, to find the most efficient coordination mechanism.

3.2 The CoPS process

The CoPS process is a sub-process of the ECo process. It structures the decision making during the implementation of a coordination mechanism. The CoPS process addresses decisions on the global level, i.e. among all entities, and on the local level, for each entity individually. The CoPS process is shown in Figure 2. The global process step is the definition of commonly accepted conversation protocols. It is global in the sense that all agents have to agree on the same conversation protocols to allow for an effective coordination. All other steps of the CoPS process have to be done by each entity by itself; therefore they are referred here as local. First each entity has to define its conversation policy. A conversation policy is a ”restrictions on communication based on the content of the communicative act" [13]. Within a conversation strategy a planning entity has to encode which concessions it is willing to make to whom, for instance.

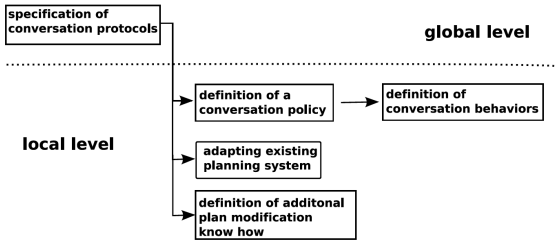


Fig. 2: Overview of the CoPS process

A conversation strategy is implemented in the conversation behaviors. A conversation behavior is executed in a particular state of the conversation, i.e. a state in the conversation automaton of one of the participants of the conversation.

The access to the local planning system can either be done directly, if the planning system is part of the agent, or by using integration techniques, like web-services, for instance.

It might be useful to add local planning-relevant knowledge to the agent, so that the agent can modify the input data of the planning system in a meaningful way. This could lead to reduced interaction times between the planning system and the agent, as the agent can modify the input data in a way that allows the planner to operate more efficiently.

3.3 The CoPS framework

The CoPS framework supports the implementation phase of the ECo process. The CoPS framework aims to facilitate the implementation. Within the CoPS framework the abstract implementation of a *planning authority agents*, the agent that represents a planning authority, and a coordination agent is provided. A coordination agent represents a network of agents, that needs to coordinate their activities and performs some management and bookkeeping actions for the entire network. The agents of the framework have to be instantiated and concrete strategies for conversations have to be implemented, as well as, the access to the planning system and additional knowledge how the planner should be used. In particular the definition and reuse of conversation protocols and their localization for each planning entity are supported by the CoPS framework.

4 Applying the ECo-CoPS approach: A case study

The ECo-CoPS approach has been applied to different case studies from the fields of logistics [18] and ambient intelligence [19]. Here we present a case study from the field of logistics. We use a simple setting within a manufacturing process of a company. First goods have to be produced, then they have to be packed and finally shipped to the customer. The work flow is detailed in Figure 3. A

Fig. 3: Work flow of a the production and distribution example

number of orders have to be satisfied. Each orders specifies a type of product, a destination for the shipment and a due date. In the beginning all orders are released and a plan to satisfy all orders has to be computed. During the production process a scheduling problem has to be solved, which has been taken from the literature [3, 5]. To compute a packing plan a 3-D bin packing problem has to be solved. Finally, to plan the shipment a vehicle routing problem has to be solved. Each of these problems is known to be a computational hard problem. We have developed three independent planning systems, each responsible for computing a valid sub-plan. An order is completed if all products are shipped to the customers. If the delivery date is later than the specified due date a penalty per time unit of lateness is imposed. We present here a compressed version of the case study, the complete case study can be found in [18].

The overall modeling of this problem is done using a set-constraint based approach, which is omitted here. During the modeling phase all relevant concepts are defined, which are necessary for the definition of the coordination requirements and to define measurements for the overall performance of the company. The coordination requirements that are of particular interest in this case study are that for each sub-problem a feasible plan exists and the overall global plan is feasible. This requires that the planning sequence is correct for all items, and that all items are produced, packed and shipped requested in the orders. As an global objective function we use the overall costs, compromising the costs of packaging, the costs for transportation, and eventually penalties for lateness.

The selection step contains two stages. A first identification step to shrink down the number of candidates and a qualitative evaluation as a second step to analyze if the candidate mechanisms satisfy the coordination requirements. In the identification step we take advantage of the idea of building repositories of mechanism description, which are annotated with relevant characteristics. Therefore we have build up a repository of different types of coordination mechanisms and classified them according to coordination specific characteristics. In brackets we point out the characteristic the scenario requires. The characteristics are presented in form of binary questions and are the following: Does an allocation problem exists? (No); Are the local objective functions comparable? (No); Are the planning systems homogeneous or heterogeneous? (heterogeneous); Does a common objective function exists? (Yes); Is information hiding necessary? (No); Do cyclic dependencies exist? (No). More details of the classification can be found in [18]. By browsing the repository of different classes of coordination mechanisms, according to the needs of the current scenario, we can restrict the number of coordination mechanisms that have to be investigated in depth for the applicability for the given scenario. As a result of the first step we identify

the following coordination approaches as possible candidates: plan merging, decentralized planning for a centralized plan, result sharing, and negotiation. Note that for this simple case the approach of decentralized planning for a centralized plan [6] is equivalent to result sharing. Different planners compute partial solution and pass them to the next planner which is then generating his part of the overall plan. This, in fact, is result sharing. The sequence of the planning systems computing their partial plan is given by the work flow presented in Figure 3. The plan merging approach requires an additional entity that collect all local plans and is capable of integrating them and, if necessary, propose plan modifications to ensure consistency. This requires planning knowledge to compute plan modifications that ensure a feasible global plan, as well as, feasibility of the local plans. Therefore this solution is similar to a complete centralized planner, which is not in the scope of this research.

The class of negotiations as coordination means cover a wide field. In this case study a key problem is that most costs are fixed in the last planning step, where the least flexibility of planning decision exists. Ideally a backward oriented planning would be more appropriate. But this approach makes it more complex to ensure feasibility of the overall plan, as the execution sequence of the planning systems would be directly inverse to the sequence imposed by the dependencies among the planning problems. A solution to this problem can be a mechanism that facilitates the exchange of requirements towards the local plans, and plan suggestions that tries to satisfy the requirements and still ensuring feasible local plans. Such a coordination approach would result in a sequence of exchanges of requirements to, and suggestions of plans. This corresponds to a negotiation, trying to minimizing the total costs. By starting with the parts of the planning process where most of the costs are fixed requirements can be identified that lead to an overall solutions with lower costs. Previous planning stages have to identify what requirements are possible to fulfill and offer those to the subsequent planning entity.

In the implementation step the remaining two major concepts, result sharing and negotiations, are implemented. Note that we have only to derive the agents from the CoPS framework, implementing the particular coordination mechanism, and enable them to use the existing planning systems. Therefore, the efforts for implementing these coordination systems are considerable low. The result sharing approach is referred here as sequential planning, as the planning steps are done sequentially, following the dependencies among the planning problems. The agents access their planning systems using web services. In the evaluation phase we compare both approaches using randomly generated problem instances of different size. First, we analyze how both approaches scale with the problem size. Second, we perform a detailed analysis for specific problem sizes. For the first analysis we consider scenarios from 1 up to 30 orders. The resulting costs for both approaches are shown in Figure 4. Note that the scales of the subfigures are not identical. We do so, to allow the reader to see the differences also between scenarios with few orders. For one order both methods are equivalent and generate the same plan. In all other scenarios the improved, negotiation-

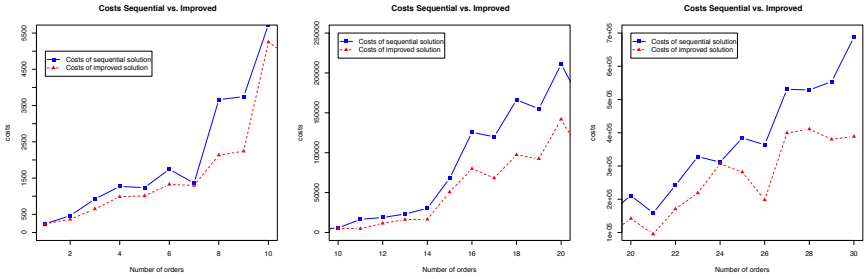


Fig. 4: Scaling of both coordination approaches with different problem sizes (1–30 orders)

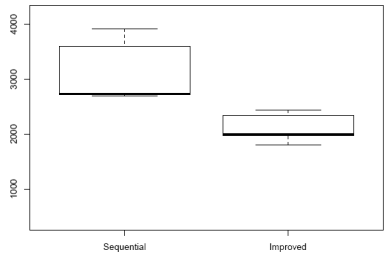
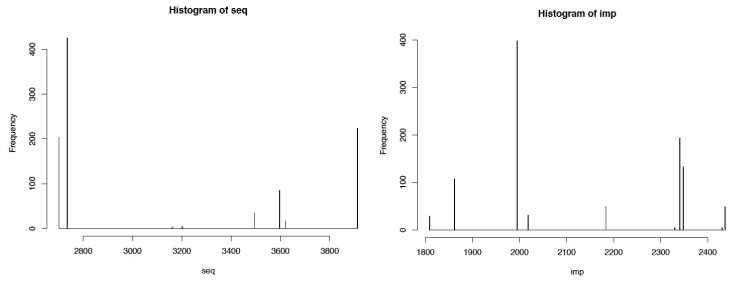


Fig. 5: Box plots for instance 2 comparing costs of the sequential and improved coordination approach

based, approach performs better than the sequential planning. Even though this data does not allow to draw a conclusion, as the number of instances is too small, it shows a clear indication. Moreover, we can see that about 7 orders the costs increase drastically. With about seven orders the first penalties have to be paid as not all orders can be performed in time. The second drastic increase can be seen at approx 15 orders. Then the system goes in an overload situation, where nearly all orders cannot be performed in time, and the penalties rise dramatically.

Based on these results we investigate particular problem sizes in more detail. In total we created 10 different scenarios consisting of the same number of orders and compute 1000 replications for each scenario. Here we present the results obtained with a scenario with five different orders. If we compare the results between both approaches we can summaries, for this scenario that the improved, negotiation based, coordination approaches, leads to a better overall performance and a more stable result, as the spread of the results is lower, than the result sharing approach. We present in Figure 5 the box plots comparing the mean costs and the spread obtained in different runs. As typically for planning systems the spread results from the fact that a few different solutions are computed over and over again. The resulting histograms for both approaches are shown in



(a) sequential (b) improved

Fig. 6: Histogram for the sequential and improved approach for instance 2

Figure 6. Summarizing this evaluation we can now select an effective and efficient coordination mechanism, as a result of applying the ECo process. For the given situation at hand the negotiation based approach has to be selected.

5 Summary and Outlook

In this paper we argued that it is necessary to provide methods for identifying effective and efficient concepts during the design of multiagent systems to support developers. We have exemplified this, addressing the selection process of coordination mechanisms for autonomous planning agents. Up to now, this problem mainly occur in logistic scenarios. With the ongoing trend towards ubiquitous intelligent system the task to coordinate intelligent planning systems is going to spread into various domains. Therefore, we have discussed the application of the ECo-CoPS approach also in the field of ambient intelligent systems [19]. We have detailed that research in the field of AOSE has covered this field not sufficiently. For that reason we have presented the ECo-CoPS approach that defines a structured process for the selection of coordination mechanisms for autonomous planning systems, where the local autonomy of the agents, as well as, the existing planning systems can be preserved. A case study has been presented to detail how the ECo-CoPS approach can support the selection process. Moreover, we think that the ECo process can be used as a blueprint for the selection of other concepts in multiagent systems, as well.

As the ECo-CoPS approach presents a process for handling specific problems its assessment becomes more sound by multiple iterations of the process. This allows for analyzing if additional tailoring of the process or the definition of additional supporting sub-processes might be useful. Therefore we strive to apply the process in more case studies from different domains. It turned out that the modeling step of the ECo process can become time intensive. Therefore we want to investigated the usage of different modeling techniques for the coordination problems. To take more advantage of the efforts in the modeling phase we

want to generate more synergies between the modeling and the implementation step. Therefore we are considering to use specific UML profiles and the object constraint language (OCL) for modeling, as this way of model might offer additional value during the implementation phase. Therefore the CoPS process and in particular the CoPS framework might have to be adapted.

Acknowledgment

This work was been supported by a fellowship within the Postdoc-Programme of the German Academic Exchange Service (DAAD).

References

1. Claudio Bartolini, Chris Preist, and Nick R. Jennings. Architecting for reuse: A software framework for automated negotiation. In John Mylopoulos, Michael Winikoff, and Nick R. Jennings, editors, *Agent-Oriented Software Engineering III Proc. of the Third International Workshop, AOSE 2002*, volume 2585, pages 88 – 100, Bologna, Italy, 2003. Springer.
2. Frances M. T. Brazier, Frank Cornelissen, Rune Gustavsson, Catholijn M. Jonker, Olle Lindeberg, Bianca Polak, and Jan Treur. Compositional design and verification of a multi-agent system for one-to-many negotiation. In *Proceedings of the Third International Conference on Multi-Agent Systems, ICMAS'98*, pages 49 – 56. IEEE Computer Society Press, 1998.
3. Robert W. Brennan and William O. A simulation test-bed to evaluate multi-agent control of manufacturing systems. In *WSC '00: Proceedings of the 32nd conference on Winter simulation*, pages 1747–1756, Orlando, Florida, 2000. Society for Computer Simulation International.
4. Stefan Bussmann, Nick R. Jennings, and Michael Wooldridge. Re-use of interaction protocols for agent-based control applications. In Fausto Giunchiglia, James Odell, and Gerhard Weiß, editors, *Agent-Oriented Software Engineering III Proc. of the Third International Workshop, AOSE 2002*, volume 2585 of *Lecture Notes in Computer Science*, pages 73 – 87, Bologna, Italy, 2003. Springer.
5. Sergio Cavalieri, Luc Bongaerts, Marco Macchi, Marco Taisch, and Jo Weyns. A benchmark framework for manufacturing control. In *2. International Workshop on Intelligent Manufacturing Systems*, pages 225 – 236, Leuven, Belgium, 1999.
6. Edmund H. Durfee. Distributed problem solving and planning. In Gerhard Weiß, editor, *Multiagent Systems: a modern approach to distributed artificial intelligence*, pages 121 – 164. MIT Press, 1999.
7. Foundations for Intelligent Physical Agents FIPA. Fipa standard specifications, 2002. `http://www.fipa.org/repository/standardspecs.html`, Accessed: 02/04/11.
8. Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of reusable object-oriented software*. Addison-Wesley Professional Computing Series. Addision Wesley Longman Inc., Reading,, 1994.
9. Luca Gardelli, Mirko Viroli, and Andrea Omicini. Design patterns for self-organising systems. In Hans-Dieter Burkhard, Gabriela Lindemann, Rineke Verbrugge, and László Z. Varga, editors, *Proc. of the 5th International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2007*, Lecture Notes in Artificial Intelligence, pages 123 – 132, Leipzig, 2007. Springer.

10. Sandra C. Hayden, Christina Carrick, and Qiang Yang. Architectural design patterns for multiagent coordination. In *Proceedings of the 3rd International Conference on Autonomous Agents, AGENTS'99*, 1999.
11. Vincent Hilaire, Olivier Simonin, Abder Koukam, and Jacques Ferber. A formal approach to design and reuse agent and multiagent models. In Michael Luck and James Odell, editors, *Agent-Oriented Software Engineering V; Proc. of the 5th International Workshop, AOSE 2004*, volume 3382, pages 142 – 157, New York, NY, USA,, 2005. Springer.
12. Catholijn M. Jonker, Jan Treur, and Pinar Yolum. A formal reuse-based approach for interactively designing organizations. In Michael Luck and James Odell, editors, *Agent-Oriented Software Engineering V; Proc. of the 5th International Workshop, AOSE 2004*, volume 3382, pages 221 – 237, New York, NY, USA,, 2005. Springer.
13. Lalana Kagal and Tim Finin. Modeling conversation policies using permissions and obligations. In Rogier M. van Eijk, Marc-P. Huget, and Frank Dignum, editors, *AAMAS 2004 Workshop on Agent Communication (AC2004)*, New York, 2004.
14. Manuel Kolp, T. Tung Do, and Stéphane Faulkner. Introspecting agent-oriented design patterns. In S. K. Chang, editor, *Handbook of Software Engineering and Knowledge Engineering,*, volume Vol. 3: Recent Advances, pages 151–176. World Scientific Publishing Co, 2005.
15. Jürgen Lind. Patterns in agent-oriented software engineering. In Fausto Giunchiglia, James Odell, and Gerhard Weiß, editors, *Agent-Oriented Software Engineering III, 3. International Workshop, AOSE 2002*, volume 2585 of *Lecture Notes in Computer Science*, pages 47 – 58, Bologna, Italy, 2003. Springer.
16. Andrea Omicini, Alessandro Ricci, Mirko Viroli, Cristiano Castelfranchi, and Luca Tummolini. Coordination artifacts: Environment-based coordination for intelligent agents. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, volume Volume 1, pages 286–293, New York, New York, 2004. IEEE Computer Society.
17. Sylvain Sauvage. Design patterns for multiagent systems design. In *Proceedings of the 3rd International Conference on Artificial Intelligence, MICAI'04*, volume 2972 of *Lecture Notes in Artificial Intelligence (LNAI)*. Springer, 2004.
18. R. Schumann. *Engineering Coordination : A Methodology for the Coordination of Planning Systems*. PhD thesis, Institute of Informatics, Goethe University, 2010. `http://publikationen.ub.uni-frankfurt.de//frontdoor.php?source_opus=8143`, Accessed: 02/04/2011.
19. René Schumann. Engineering coordination in future living environments. In Ralf Dörner and Detlef Krömker, editors, *Proceedings of the ITG / GI Workshop on Self-Integrating Systems for Better Living Environments 2010: SENSYBLE 2010*. Shaker Verlag, 2011. accepted for Postproceedings.
20. Tom De Wolf and Tom Holvoet. Design patterns for decentralised coordination in self-organising emergent systems. In Sven A. Brueckner, Salima Hassas, M·rk Jelasity, and Daniel Yamins, editors, *Engineering Self-Organising Systems: Proceedings of the 4th InternationalWorkshop, ESOA 2006*, volume 4335 of *Lecture Notes in Artificial Intelligence*, pages 28 – 49, Hakodate, Japan, 2006. Springer.