Agent Contest Competition: 4th Edition

Tristan M. Behrens[2], Mehdi Dastani[1], Jürgen Dix[2] and Peter Novák[2]

[1]Utrecht University
P.O.Box 80.089, 3508 TB Utrecht, The Netherlands
mehdi@cs.uu.nl
[2]Clausthal University of Technology
Julius-Albert-Str. 4, 38678 Clausthal-Zellerfeld, Germany
{tristan.behrens|dix|peter.novak}@tu-clausthal.de

Abstract. This paper summarises the Agent Contest 2008, organised in association with ProMAS'08. The aim of the contest is to stimulate research in the area of multi-agent systems by identifying key problems and collecting suitable benchmarks that can serve as milestones for evaluating new tools, models, and techniques to develop multi-agent systems. The first two editions of this contest were organised in association with CLIMA conference series and the third edition was organised in association with ProMAS'07. Based on the experiences from the previous three editions ([16,17,18]), the contest scenario has been changed to test the participating multi-agent systems on their abilities to *coordinate* and *cooperate*. We wanted to emphasise *team work* and *team strategy* issues in a dynamic environment where teams compete for the same resources. Seven groups from Iran, Ireland, England, France, Germany, Poland, and Turkey did participate in this years contest.

1 Introduction

Multi-agent systems are beginning to play an important role in today's software development. In the field of agent-oriented software engineering, various multi-agent system development methodologies have been proposed. Each methodology focuses on specific stages of the multi-agent system development. For example, Gaia [21] and Prometheus [20] focus on the specification and design stages assuming that other stages such as requirement and implementation are similar to corresponding stages of other software development paradigms. Therefore, software developers using Gaia and Prometheus propose models to specify and design multi-agent systems, while ignoring the implementation models.

Moreover, there is a growing number of agent-oriented programming languages and development platforms that are proposed to facilitate the implementation of multi-agent systems [11,15]. These programming languages and platforms introduce programming constructs that can facilitate efficient and effective implementation and execution of multi-agent systems. The development of multi-agent systems requires efficient and effective solutions for different problems which can be divided into three classes: Problems related to (1) the development of individual agents, (2) the development of coordination and

cooperation mechanisms to manage the interactions between individual agents and team work, and (3) the development of the shared environment in which agents perform their actions.

Typical problems related to individual agents are how to specify, design and implement issues such as *autonomy*, *pro-active/reactive behaviour*, *perception and update of information*, *reasoning and deliberation*, and *planning*. Typical problems related to the interaction of individual agents are how to specify, design and implement issues such as *communication*, *coordination*, *cooperation*, *negotiation*, and *team working*. Finally, typical problems related to the development of their environment are how to specify, design and implement issues such as *resources and services*, agents' access to *resources*, active and passive *sensing* of the environment, and realizing the *effects of actions*.
This competition started as an attempt to stimulate research in the area of multi-agent systems by

1. *identifying key problems in developing multi-agent systems, and*
2. *evaluating state-of-the-art tools, models, and techniques in the field of multi-agent systems.*

While there already exist several competitions in various areas of artificial intelligence (theorem proving, planning, Robo-Cup, Games, etc.) and, lately, also in specialised areas in agent systems (Trading Agent Competition (TAC) [1] and AgentCities competitions [2]), the emphasis of this contest is on the use of existing tools, models, and techniques that are proposed to develop multi-agent systems ([11,10,12,13,14,19]. In particular, we aim at evaluating existing approaches for the development of multi-agent systems where individual agents cooperate with each other to solve a task. In this respect, issues such as team working, team strategy, interaction with dynamic environment, modeling the environment, limited perception, uncertain action effects, reasoning and planning, and learning are essential.

The previous editions of this contest were organised in cooperation with CLIMA and ProMAS workshop series. The scenario from this year is changed in order to put the participating multi-agent systems under a test with respect to coordination, cooperation, and team working issues in a dynamic environment where teams of agents compete for the same resources.

2 Scenario Description

The competition task consisted of developing a multi-agent system to solve a cooperative task in a dynamically changing environment. The environment of the multi-agent system (see also [9]) is a grid-like world where agents can move from one cell to a neighbouring cell. In this environment, herds of cows can appear and move around in the environment showing swarm-like behavior. Participating agent teams are expected to explore the environment, avoid obstacles and compete with another agent team to get most cows. The agents of each team can coordinate their actions in order to control the movement of herds and move

as much cows as possible to their own corral. Agents have only a local view on their environment, their perceptions are incomplete, and their actions can fail.

There were seven teams participating in the competition:

- Jason from the ENS Mines of Saint Etienne, France, and University of Durham, UK,
- SHABaN from the Iran University Of Science and Technology,
- Jadex from the Hamburg University of Applied Sciences,
- Bogtrotters from the University College Dublin, Ireland,
- Krzaczory from the, Polish Academy of Sciences,
- KANGAL from the Bogazici University, Istanbul, Turkey, and
- JIAC-TNG from the Technische Universität Berlin, Germany.

Each team competed against all other teams in a series of matches in parallelised tournaments on three servers. Each match between two competing teams consisted of three simulations. A simulation between two teams was a competition between them with respect to a certain starting configuration of the environment. Winning a simulation yielded three points for the team, a draw was worth one point and a loss resulted in zero points. The winner of the whole tournament was evaluated on the basis of the overall number of collected points in the matches during the tournament. In the case of an equal number of points, the winner would have been decided on the basis of the absolute number of collected cows. Details on the number of simulations per match and the exact structure of the competition has been published prior to the Contest on the official Agent Contest 2008 website at `http://cig.in.tu-clausthal.de/agentcontest2008/`.

2.1 Technical Description of the Scenario

In the contest, the agents from each participating team were *executed locally* (on the participant's hardware) while the simulated environment, in which all agents from competing teams performed actions, was run on the *remote contest simulation server* run by the contest organisers. The interaction/communication between agents from one team were managed locally, but the interaction between individual agents and their environment (run on the simulation server) took place via Internet. Participating agents were connected to one of the simulation servers that did provide the information about the environment. Each agent from each team connected to and communicated with the simulation server using TCP protocol and messages in XML format.

During the initial phase[1] agents from all competing teams connected to the simulation servers, identified and authenticated themselves and got general match information. At the announced start time of the tournament, the simulation servers were on-line and the agents from participating teams were able to connect to it. After a successful initial handshake during which agents identified

[1] The contest organisers contacted participants before the actual tournament and provided them the IDs necessary for identification of their agents for the tournament.

themselves by their IDs and received acknowledgment from the servers, they waited for the simulation start. The initial connecting phase took a reasonable amount of time in order to allow agents to be initialised and getting connected (15 minutes).

The simulation servers controlled the competitions by selecting the competing teams and managing the matches and simulations. In each simulation, a simulation server, in a cyclic fashion, provided sensory information about the environment to the participating agents and expected their reactions within a given time limit. Each agent reacted to the received sensory information by indicating which action (including the *skip action*) it wants to perform in the environment. If no reaction was received from the agent within the given time limit, the simulation server assumed that the agent performed the *skip* action. Agents had only a local view on their environment, their perceptions were incomplete, and their actions can fail. After a finite number of steps the simulation server stopped the cycle and participating agents received a notification about the end of a simulation. Then the server started a new simulation possibly involving the same teams.

2.2 Team, Match, and Simulation

An agent team consisted of six software agents with distinct IDs. There were no restrictions on the implementation of agents, although we encouraged the use of approaches based on state-of-the-art tools, methodologies and languages for programming agents and multi-agent systems, as well as the use of computational logic based approaches. The tournament consisted of a number of matches. A match was a sequence of simulations during which two teams of agents competed in several different settings of the environment. For each match, the server 1) picked two teams to play it and, subsequently, 2) started the first simulation of the match. Each simulation in a match started by notifying the agents from the participating teams and sending them the details of the simulation. These included for example the size of the grid, the corral position, the number of steps the simulation will perform, etc. A simulation consisted of a number of simulation steps. Each step consisted of 1) sending a sensory information to agents (one or more) and 2) waiting for their actions, and 3) processing agents' replies and calculating the next state of the environment. As mentioned above, in the case that an agent did not respond within a timeout (specified at the beginning of the simulation) by a valid action, it was considered to perform the *skip* action in the given simulation step.

2.3 Environment objects

The (simulated) environment was a rectangular grid consisting of cells. The simulated environment contained two corrals—one for each team—which serve as a location where cows should be directed to. Each cell could contain either nothing, an agent, a cow or an obstacle. If a cow entered a corral it was removed.

Agents could enter the corrals without effect. All three maps were hand crafted for the particular scenario.

2.4 Actions and perceptions

At the start of each simulation the agents received the details of the environment:

- simulation ID,
- opponent's ID,
- grid size,
- corral position and size, and
- number of steps the simulation will last.

Agents were located in the grid and the simulation server provided each agent with the following information in each step:

- information about the cells in the visibility range of the agent (including the one agent stands on),
- the agent's absolute position in the grid,
- the current simulation step number,
- the number of caught cows and
- the deadline for responding.

If two agents were standing in each other's field of view, they were able to recognise whether they are enemies, or they belong to the same team. Also, individual cows were identifiable.

All perceptions except for the agent's and the corral's position were subject to be "forgotten" by the server, whereas the server never gave wrong information.

Agents were allowed to perform one action in a simulation step. The following actions were allowed:

- *skip* – the agent does nothing,
- *north* – the agent moves to the north,
- *northeast* – the agent moves to the northeast,
- *east* – the agent moves to the east,
- *southeast* – the agent moves to the southeast,
- *south* – the agent moves to the south,
- *southwest* – the agent moves to the southwest,
- *west* – the agent moves to the west,
- *northwest* – the agent moves to the northwest.

All actions, except the *skip* action, could fail. The result of a failed action is the same as the result of the *skip* action. An action can fail either because the conditions for its successful execution are not fulfilled or because of the information distortion.

2.5 Cow Movement Algorithm

Cows are simple creatures. They tend to move away from cells that they do not like and to move towards cells they do like. Cows want to move away from agents and trees. On the other hand, they are attracted by empty spaces and they want to stay close to other cows, however not too close. Cows have the tendency to form herds, which tend to be tighter in times when the animals are scared by cowboys.

The cows have two fixed visibility ranges. Cows are attracted to other cows that are in the visibility-square and not too close and they are repelled by cows that are too close.

Cows are slower than agents. Each cow only moves every three steps. Our simulation ensures that all cows do not move in the same step using a simple algorithm.

The direction in which a cow will move in the next step is determined by calculating a weighted linear-combination of the distance-vectors to visible cells, with weights respective to the content of the cells. Cows do not move if the resulting vector is zero. See [9] for technical details about cow movements.

2.6 Final Phase of the simulation

In the final phase, the simulation server sent a message to each agent allowing them to disconnect from the server. By this, the tournament was over.

3 Submission

The participation in this contest consisted of two parts. Participants first submitted the description of analysis, design and implementation of a multi-agent system for the above application. We encouraged the use of existing state-of-the-art multi-agent system methodologies to describe the systems. For the description of the implementations, the participants were asked to explain how the design is implemented. This could be done by explaining, for example, which programming language, platform, tools, and techniques are used to implement the multi-agent system. All teams, except the one from Turkey, provided submissions that are included in this volume.

The second part of the contest was the actual participation in the tournament by means of an (executable) implementation of a multi-agent system. The agents from each participating systems (agent teams) were executed locally (on the participant's hardware) while the simulated environment, in which all agents from competing teams perform actions, was run on a remote contest simulation server. Interaction/communication between agents from one team has been managed locally, but the interaction between individual agents and their environment (run on the simulation server) was via Internet.

3.1 Received Submissions

For the 2008 edition of the Contest we initially received 9 submissions from 7 countries from all around the globe with a majority from Europe: JIAC-TNG [5] (Germany), Jadex [6] (Germany), SHABaN [7] (Iran), Krzacory [3] (Poland), Jason [8] (France/United Kingdom), Bogtrotters [4] (Ireland), KANGAL (Turkey), FLUX (Germany) and CSIRO (Australia). Shortly before the Contest launch, the teams CSIRO and FLUX withdrew due to technical and organizational issues in the development team, thus leaving finally 7 teams to compete in the Contest. Detailed descriptions of the submissions (except for KANGAL team) are included in this volume.

In comparison to the last editions, in this year's Contest we could observe a rise of using more formal approaches to system analysis and design. Four teams (JIAC-TNG, Jadex, Jason and Bogtrotters) used a state-of-the-art methodology to devise the multi-agent system architecture of their team. One team (SHABaN) used a MAS prototyping language to evaluate their early designs. Finally the teams KANGAL and Krzacory used either ad-hoc design, or their approach was partly based on a utility function optimization technique.

Almost all the teams came up with a design using two generic role types for their agents: *herders* and *explorers*. However, the resulting designs differ in coordination techniques as well as approaches to MAS organisation and role-assignment. According to the agent coordination the approaches can be divided into two groups: those using a rather *decentralised* approach (JIAC-TNG, Jason and Bogtrotters) and teams with a single *centralised* coordination entity/agent (Jadex, SHABaN and Krzacory).

The centralised approaches used the main coordinator/master agent for steering the agents in the teams, however it can be observed that anyway all these approaches left a significant part of the autonomous acting and decision making on single agents (e.g. obstacle avoidance, exploration strategy, etc.). Unlike in the previous Contest editions, this year we did not see a truly centralised approach - a one in which agents lack autonomy and are completely directed by the team managing agent.

The approaches employed by the teams without a centralised control and MAS organisation varied from using auctions for assignment a role in a team to particular agents (Bogtrotters) to sharing intentions among agents in a team (JIAC-TNG).

We observe also an interesting trend in approaches to agent navigation in the environment. It seems that more and more teams employ the A^* algorithm to search for shortest paths in the map of the environment. Thus the navigation in even complex environments is not that much of an issue as we could observe in previous Contest editions.

Another interesting arising trend seems to be employment of MAS recovery monitoring mechanisms to keep the agent team up and running. As this used to be an issue in the previous years of the Contest, the teams JIAC-TNG and Bogtrotters implemented a team recovery technique to restart/recreate a crashed

agent as well as to inform the restarted agent about the current status of the team knowledge (note that both teams use a decentralised approach).

4 Technical infrastructure

In the fourth edition of this Agent Contest, we re-used the technical infrastructure we developed for the previous editions. Briefly, the server's architecture consists of

1. *simulation plug-in*: A replaceable module providing the logics of the environment simulation,
2. *agent session manager*: Responsible for holding the sessions between the server and individual agents and en/de-coding of XML messages of the protocol,
3. *visualization library*: It produced the SVG records from each time frame of the simulation environment state,
4. *contest webinterface*: Providing a public view and interface to the MASSim server, and
5. MASSim core module: Managing the tournament scheme and providing the connection between the simulation plug-in, agent session manager and web-interface.

A more detailed description of the system can be found in the report on the second edition of the Agent Contest [17]. The system is published on the official Contest website: `http://cig.in.tu-clausthal.de/AgentContest/`.

4.1 Contest preparation

As in previous editions, before the tournament itself, the Contest organisation went through several preparatory stages. We released the scenario description for the Agent Contest on 18 February 2008 and updated on 18 April 2008. The communication protocol for the simulation scenario was released later on 13 March 2008. The Agent Contest testing phase was launched on 29 April 2008 and ran until the very Contest tournament launch on 26 May 2008. During this period, which lasted more than one month, the participants could freely connect to the testing server and test their agents in a simulated match against our dummy *Bot* agent team. We did not allow different teams to compete against each other as this should happen only during the tournament itself. During the testing phase, few minor bugs in the scenario implementation were discovered and quickly fixed.

4.2 Tournament

The Agent Contest 2008 tournament itself was launched on Monday, May 26th 2008 at about 10:00 CEST (UTC/GMT+2). A few days in advance, the participants received the Internet coordinates of the tournament server together with

credentials for their agents. The Agent Contest was served on the three tournament servers called *Agent-Contest1*, *Agent-Contest2*, and *Agent-Contest3* that could be observed via a web-interface at the address `http://agentserver.in.tu-clausthal.de`. We provided also a chat space for participants, what in the course of the tournament itself turned out to be a vital and efficient communication tool.

The teams competed against each other on four successive days and based on three different simulation servers. The time table of these matches are shown below[2]:

Day \ server	Agent-Contest1	AgentContest2	AgentContest3
26th May	Jason vs SHABaN	Jadex vs Bogtrotters Jadex vs KANGAL Bogtrotters vs KANGAL	JIAC-TNG vs krzaczory
27th May	Jason vs Jadex	JIAC-TNG vs Bogtrotters	krzaczory vs KANGAL krzaczory vs SHABaN KANGAL vs SHABaN
28th May	Jason vs krzaczory Jason vs Bogtrotters krzaczory vs Bogtrotters		JIAC-TNG vs Jadex JIAC-TNG vs SHABaN Jadex vs SHABaN
29th May	Jason vs JIAC-TNG Jason vs KANGAL JIAC-TNG vs KANGAL	krzaczory vs Jadex	SHABaN vs Bogtrotters

All results, together with the SVG recordings of all the matches can be downloaded from `http://agentserver.in.tu-clausthal.de`.

4.3 Simulation instances

The teams competed in matches each consisting of 3 different grid simulations with identifiers *CowSkullMountain*, *RazorEdge* and *Street* (Figure 1). All scenarios are handcrafted labyrinths to challenge agent teams obstacle avoiding and communication approaches.

5 Contest results

The winner of the ProMAS Agent Contest 2008 was the JIAC-TNG team from the DAI-Labor, Technische Universität Berlin, Germany. They gained the highest number of points: 46. The second team was Jadex (Germany) with 42 points followed by the SHABaN team (Iran) with 37 points. The summary of the whole tournament is summarised in the Table 1.

6 Conclusion

As in the previous Contest editions, our main motivations behind this Agent Contest are the following:

[2] The table is fragmented due to the fact that the tournament was originally scheduled for 9 participating teams.

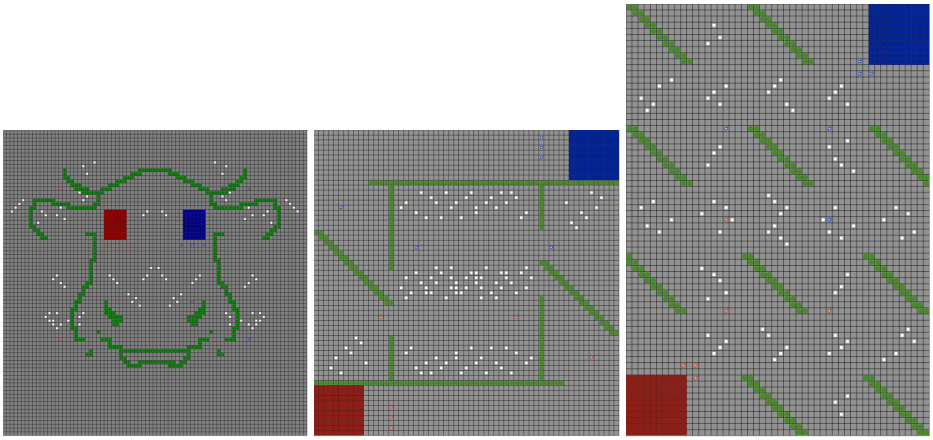

Fig. 1. Initial simulation scenarios *cowskullmountain*, *razoredge*, and *street*

Rank	Team	CowScore	Points
1.	JIAC-TNG team	643	64
2.	Jadex	542	42
3.	SHABaN	373	37
4.	krzaczory	379	26
5.	Jason	393	21
6.	bogtrotters	305	13
7.	KANGAL	32	1

Table 1. Final tournament results.

- to foster the research and development of practically oriented approaches to programming multi-agent systems, and
- to evaluate the state-of-the-art techniques in the field, and
- to identify key problems using these techniques.

After the success of the previous three editions of the Agent Contest we recognised a need to shift the main focus of the Contest scenario from basic agent-system issues (testing the state-of-the-art approaches to programming agents) more towards a *multi*-agent setting, i.e. coordination and cooperation strategies among agents in a MAS team. For the 2008 edition we devised a new scenario *cows & cowboys*, which turned out to be more challenging and entertaining than the previous *gold miners* scenario. The main emphasis was to construct a competition scenario in such a way that the success of the team *should strongly depend on coordination of several agents*. This was achieved by our design decision, not to allow to push a group of cows in a certain direction by a *single agent*.

Although initially we have been rather sceptical about solubility of the scenario (and we still do not know a perfect solution), it turned out that the competing teams performed rather well. The most difficult scenario turned out to be the *RazorEdge* map 1. To push a group of cows through the narrow opening in

the map so that cows do not escape in the wrong direction turned out to require good cooperation abilities of the agent team. In scenarios similar to this we see still a potential for improvement of agent team performance.

Similarly to the previous Contest editions, we collected interesting feedback from the participants. To our pleasure, it turns out that one of the main gains from participating in the Agent Contest tournaments are contributions to testing and debugging of the participants MAS-oriented frameworks and programming systems. Another important aspect seems to be the educational value of the Contest: We seem to attract more and more teams including students on both post-graduate, as well as undergraduate levels.

We run this year's Contest edition in a different organisational structure. We divided the tournament into four sub-tournaments, each ran on a separate day. On each day we executed three parallel contests. This resulted into a significant decrease of the tournament running time and allowed us to use larger maps and more complex scenarios for individual simulations. In the future we want to further follow this line.

7 Acknowledgements

We are very thankful to the students of the Department of Informatics of Clausthal University of Technology. They worked very hard in order to meet all the deadlines and deliver high-quality code. In particular, our thanks go this year to

- *Jens Dehnert* and
- *Slawomir Deren*

for the numerous hours they have invested to help us get the scenario and the tournament ready in time.

References

1. `http://www.sics.se/tac`.
2. `http://www.agentcities.org/EUNET/Competition`.
3. AC08 system description. Sixth International Workshop on Programming Multi-Agent Systems, 2008.
4. Dublin Bogtrotters: Agent Herders. Sixth International Workshop on Programming Multi-Agent Systems, 2008.
5. Herding agents - JIAC TNG in Multi-Agent Programming Contest 2008. Sixth International Workshop on Programming Multi-Agent Systems, 2008.
6. On Herding Artificial Cows: Using Jadex to Coordinate Cowboy Agents. Sixth International Workshop on Programming Multi-Agent Systems, 2008.
7. SHABaN multi-agent team to herd cows. Sixth International Workshop on Programming Multi-Agent Systems, 2008.
8. Using *Jason* and $\mathcal{M}$oise$^+$ to develop a team of cowboys. Sixth International Workshop on Programming Multi-Agent Systems, 2008.

9. T. M. Behrens, M. Dastani, J. Dix, and P. Novák. Technical aspects of the agent contest competition: 4th edition. Technical Report IfI-08-05, Clausthal University of Technology, Dept of Computer Science, to appear 2008.
10. R. H. Bordini, M. Dastani, J. Dix, and A. E. Fallah-Seghrouchni, editors. *Programming Multi-Agent Systems, First International Workshop, ProMAS 2003, Melbourne, Australia, Revised and Invited Papers*, volume 3067 of *Lecture Notes in Computer Science*. Springer, 2004.
11. R. H. Bordini, M. Dastani, J. Dix, and A. E. Fallah-Seghrouchni, editors. *Multi-Agent Programming: Languages, Platforms and Applications*, volume 15 of *Multi-agent Systems, Artificial Societies, and Simulated Organizations*. Springer, Berlin, 2005.
12. R. H. Bordini, M. Dastani, J. Dix, and A. E. Fallah-Seghrouchni, editors. *Programming Multi-Agent Systems, Second International Workshop, ProMAS 2005, New York, USA, Revised and Invited Papers*, volume 3346 of *Lecture Notes in Computer Science*. Springer, 2005.
13. R. H. Bordini, M. Dastani, J. Dix, and A. E. Fallah-Seghrouchni, editors. *Programming Multi-Agent Systems, Third International Workshop, ProMAS 2005, Utrecht, The Netherlands, Revised and Invited Papers*, volume 3862 of *Lecture Notes in Computer Science*. Springer, 2006.
14. R. H. Bordini, M. Dastani, J. Dix, and A. E. Fallah-Seghrouchni, editors. *Programming Multi-Agent Systems, Fourth International Workshop, ProMAS 2006, Hakodate, Japan, Revised and Invited Papers*, volume 4411 of *Lecture Notes in Computer Science*. Springer, 2007.
15. R. H. Bordini, M. Dastani, J. Dix, and A. E. Fallah-Seghrouchni, editors. *Multi-Agent Tools: Languages, Platforms and Applications*. Springer, Berlin, to appear 2009.
16. M. Dastani, J. Dix, and P. Novák. The First Contest on Multi-Agent Systems based on Computational Logic. In F. Toni and P. Torroni, editors, *Proceedings of CLIMA '05, London, UK*, volume 3900 of *Lecture Notes in Artificial Intelligence*, pages 373–384. Springer, Berlin, 2006.
17. M. Dastani, J. Dix, and P. Novák. The second contest on multi-agent systems based on computational logic. In K. Inoue, K. Satoh, and F. Toni, editors, *CLIMA VII*, volume 4371 of *Lecture Notes in Computer Science*, pages 266–283. Springer, 2006.
18. M. Dastani, J. Dix, and P. Novák. Agent Contest Competition: 3th edition. In M. Dastani, A. El Fallah Seghrouchni, A. Ricci, and M. Winikoff, editors, *Proceedings of ProMAS'06, Honolulu, Hawaii*, volume 4908 of *Lecture Notes in Artificial Intelligence*, pages 221–240. Springer, Berlin, 2007.
19. M. Dastani, A. E. F. Segrouchni, A. Ricci, and M. Winikoff, editors. *Programming Multi-Agent Systems, Third International Workshop, ProMAS 2007, Honolulu, USA, Revised and Invited Papers*, volume 4908 of *Lecture Notes in Computer Science*. Springer, 2008.
20. L. Padgham and M. Winikoff. Prometheus: A methodology for developing intelligent agents. In *Agent-Oriented Software Engineering III: Third International Workshop (AOSE'02)*. Springer, LNAI 2585, 2003.
21. F. Zambonelli, N. R. Jennings, and M. Wooldridge. Developing multiagent systems: The Gaia methodology. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 12(3):317–370, 2003.