AgentSteel: An agent-based Online System for the Planning and Observation of Steel Production

Sven Jacobi, Cristián Madrigal-Mora, Esteban León-Soto, Dr. Klaus Fischer

DFKI GmbH
Stuhlsatzenhausweg 3
66123 Saarbrücken

Germany

{Sven.Jacobi, Cristian.Madrigal, Esteban.Leon, Klaus.Fischer} @dfki.de

ABSTRACT

The steel production of the German company Saarstahl AG, a global respected steel manufacturer, represents a Supply Chain which comprehends several time-critical and highly interference susceptible processes. Therefore, flexibility, robustness and fast reorganisation are indispensable requirements on a system responsible for the planning of production inside this Supply Chain to ensure acceptable costs and retain their respected position on this market. In this paper we present an agent-based generic solution for the planning and observation of the complete production process inside the steelwork Völklingen of Saarstahl AG. The presented system calculates solutions for given daily target schedules – based on the concrete orders by the customers – by a combined distributed online planning and scheduling algorithm, and, additionally, performs the observation of its realisation. Moreover, the presented real-time multiagent system supports the reorganisation after operational faults in this high dynamic context by suggesting new solutions to the responsible conductor.

Categories and Subject Descriptors

D.2.11 [Software Architectures]: *Domain-specific architectures*

General Terms

Algorithms, Management

Keywords

Multiagent System, Online Planning, Online Scheduling, Service Oriented Architecture

1. INTRODUCTION

In the permanent ongoing production of Saarstahl AG, a variable set of uncertainties have a strong influence on the current planning. Therefore, flexibility and fast reorganization after such interferences at any point is needed to ensure a production without interruptions which still meets the needs of the customers at acceptable costs.

The Supply Chain of Saarstahl AG as depicted in figure 1 consists of a furnace factory from which the pig iron needed as the fundamental ingredient for steel is produced. Inside so called tornados, the pig iron is brought by railway to the steelwork in Völklingen. Here, the steel of various qualities, based on the concrete needs of the customers, is poured into blocks. In the next step at the masticators, these steel blocks are manufactured to the specialised orders of the customers, i.e. to steel wire, sheets or screws mostly for the automotive cluster, aeroplane or ship building companies with their high quality requirements to these products. From here the customers are delivered with their ordered goods. In between these production steps there are more than 50 storages for which an inventory management is needed.

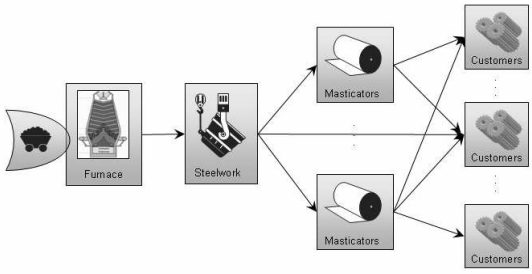

Figure 1: Supply Chain of Saarstahl AG

The system described in this paper concentrates on the production of steel inside the steelwork. The steelwork has been identified as the bottleneck of the complete Supply Chain. We describe how the system is integrated into the IT-environment of the complete Supply Chain of Saarstahl; however, this paper focuses on the identified online planning and scheduling problem inside the steelwork and its interfaces inside the complete production.

The steelwork itself consists of three converters with their corresponding flushing aggregates – together the so called converter pool – where the delivered pig iron is filled in and, by blowing oxygen into the iron, the coal content is decreased to receive steel of lower quality. Then, following the direction of production, there is the so called Secondary Metallurgy. This part of the steelwork consists of five distinct aggregates, in which certain ingredients like chrome or sulphur are added to achieve the demanded quality of steel. At the end, there are the pouring aggregates where the steel is poured into blocks of various kinds for further manufacturing in the masticators; the particular steelwork has five pouring aggregates with connected towers in which following charges in the pans can be stored until they are needed. Figure 2 summarises the structure of the steelwork. Inside the steelwork, the steel is transported in so called pans, which are able to transport steel up to 170 tons; this amount is called "charge". The pans have certain constraints on the steel they can be filled with. For example, a pan containing a charge with chrome cannot transport a charge containing no chrome next. These charges are poured in sequential order grouped in so called sequences of charges with similar qualities. These sequences have to be poured without any interruptions to guarantee the required qualities.

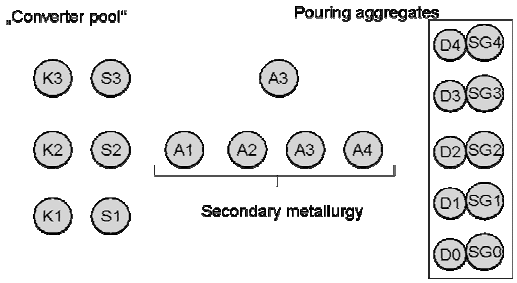

Figure 2: Steelwork "Völklingen"

Having shortly described the structure of the Supply Chain and the steelwork, we now want to model the problems we are faced during the production and describe their solutions.

In Section 2 we describe and analyse the resulting problems we are faced, in Section 3 we describe their solutions in detail. Section 4 is dedicated to the chosen agent architecture and internal architecture of the implemented system as well the external architecture regarding the embedding into the IT-environment of Saarstahl. In Section 5 we give an overview of the current work about the extension of the system as well as future work.

2. PROBLEM DESCRIPTION

Based on the customer requirements and the concrete orders to the Saarstahl AG, the company has to configure daily target schedules for the steel production inside the steelwork and observe their realisations.

A daily target schedule consists of a total ordered set of sequences for each pouring aggregate for the next 48 hours and has to fit into the actual state of the steelwork. It is refreshed usually every morning and in case of urgent new planning it can be changed any time. Hence, the daily target schedule consists of a partial ordered set of sequences concerning all pouring aggregates together because of their parallel production at the pouring aggregates. A sequence consists of a total ordered set of charges whereas the number of charges is limited by seven for each sequence. Hence, a daily target schedule consists of a partial ordered set of charges which have to be produced. A charge again consists in dependency of its quality of a total ordered set of working steps which have to be fulfilled at determined aggregates of steelwork to achieve the required quality. Once a sequence has start pouring, it requires the following charge to be at the pouring aggregate in time, because a sequence may not be poured with any interruption, otherwise the complete sequence fails.

A lot of uncertain changing circumstances like pig iron supply from the furnaces or changing orders by the masticators and their following customers in the Supply Chain demand a continuous adjustment in the planning process. The pig iron supply defines the amount and quality of potential steel to produce. Because of the fact, that the pig iron supplier does not only serve this steelwork but also others, the amount and also very important the arrival time by railway has to be negotiated. A sequence cannot be produced, if not enough pig iron available in time can be guaranteed. On the other hand, the ongoing and not determined incoming orders by the customers demand a continuous change of these daily target schedules. A charge consists of about 170 tons of steel, an order of a customer might consist of any amount. Hence, one charge does not necessarily serve one order with its specified requests. The "highest" quality requirement of an order therefore defines the quality of the complete charge; to save costs it is necessary to readjust these daily target schedules by keeping the qualities as low as possible, but still meet all requirements.

These are only two examples of influences from the outside which underline the dynamic environment of the system. The production process itself inside the steelwork again has a lot of uncertainties which make the rescheduling of the complete process too complex for human observer. For example, a time delay at a certain aggregate may have influence some hours later at another aggregate which might not be visible at first sight.

The problem the system has to deal with is, on the one hand, an online job shop scheduling problem for the aggregates involved in the production of steel inside the steelwork; on the other hand, we are faced with an online planning problem above that concerning the pans. The pans are required to be in certain specified states at certain points in time, in order to be able to execute certain tasks which are necessary to produce the demanded qualities. This immensely increases the complexity of the complete problem. These two problems influence each other; a delay of the schedule at an aggregate has impact on the availability of that certain pan for the next charge planned into this pan. Additionally the number of pans is fixed and all pans have to be used in specified time windows. The steel production is an accident sensitive process and therefore it is necessary to have a system which is able to detect potential problems as soon as possible and, moreover, which is able to handle these operational faults and return to normal business.

Even the normal scheduling inside the steelwork would be too complex to be observed optimally by a human in respect to the dynamic and fault-prone environment. Any time delay during the production must be handled very fast, because the steel coming

out of the converter has a certain calculated temperature. It cannot remain too long in the "Secondary Metallurgy", because otherwise it is too cold to be poured. Taking also the changing circumstances of the pig iron supply and the changing orders into account, the situation becomes much too complex to be handled optimally by a human.

The system has to monitor the actual states of production and analyse these data. Because of the complex and dynamic process, a lot of uncertainties arise during the production which might have influence on the process only in a few hours but already are caused now – especially concerning the planning of the pans, the system has to detect these potential errors as soon as possible and give suggestion how to reorganise and optimise the production under these new circumstances.

3. SOLUTION

In this section we describe the solution in detail. The initial input is the daily target schedule which the system may not alter first. Additionally, the actual state of production is needed to fit the calculated initial solution into the running process.

For each pouring aggregate we receive a total ordered set of sequences which consist of a total ordered set of charges and each charge has a fixed order of tasks at specified aggregates of the steelwork. For each task we know the expected amount of needed working time. Because of the reason that once a sequence has started pouring cannot be interrupted, the pouring aggregate demands the following charge at a fixed time. These times are propagated back through the aggregates of the secondary metallurgy until we reach the converter. There we have to specify the time when the blowing of oxygen into the iron has to begin. This is constrained by the fact that we are only able to produce a charge every 40 minutes in one converter; with only two converters running at the same time we are able to produce a charge every 20 minutes if we would distribute equally.

The fixed blowing times at the converter have a specified free float time for each charge. Including this float time we propagate the calculated times back to the pouring aggregates. We receive a time window with an earliest and latest start time for each aggregate. This is done with every charge of the daily target schedule. In order not to receive errors with every new received data from the steelwork, an exact calculated time is not used but a time window in which the certain step at the specified aggregate with known duration has to take place. With these time windows we also obtain more flexibility regarding the complete schedule of a charge.

Finally we receive the set of tasks for each aggregate. They locally as an agent calculate the complete set of possible schedules regarding their local objective function which might be changed by the user. Another agent receives the proposed schedules – one of each agent – and optimises them in respect to another user determined objective function for the global process for the complete steelwork. The result is the initial overall schedule. The described multiagent system [1], [2] consists of one agent for each corresponding physical aggregate including the pans P1-Pn for n used pans per daily target schedule and an additional so called planning agent which has certain coordination and negotiation tasks; figure 3 shows the agents as defined in the system.

So far the planning problem has not been taken into account. We have to ensure that each charge – meaning each quality – receives a compatible pan. For each quality we calculate the complete set of possible follow-up qualities inside the same pan concerning the complete target schedule. As a result we receive a compatibility matrix for all charges contained in the target schedule.

Now, we create a graph G(v,e) in which the nodes are the qualities labelled with the pan. The edges point to the possible follow-up pans meaning the possible qualities. Now, for a given number of m pans, m paths through this graph have to be found whereas no node may be separated meaning no quality is without a pan.

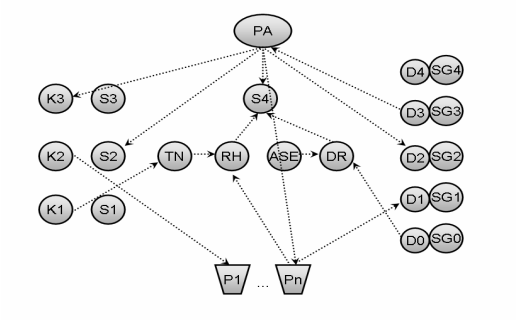

Figure 3: Agent Classification

A recursive graph normalisation in which edges are deleted by time constraints and two cases in which the results are unique is done next. In case 1, we are able to prune edges from the graph, if a node v has only one incoming edge. Then all other outgoing edges from the predecessor node can be pruned, otherwise node v might be separated. If node v has only one outgoing edge, then this path is also unique and all other incoming edges of the follow-up node can be pruned.

We then start the search PanAssignement(G'(v,e)) inside the normalised graph G'. The starting nodes are determined by the synchronisation to the actual state and the minimum of outgoing edges. For all identified starters a unique path through the graph has to be found. Having finished this, an initial solution for a daily target schedule has been calculated.

```
PanAssignment (G'(v,e))
  starter  s= pickStarterWithFewestPanOptions()
 for all pan options
    propagateOptionToOutgoingLinks(s)
     for all outgoing links
      assignPanOptionFromStarter(s)
        if (currentOutgoingLink.Count() == 0)
                node is a leaf
          if (allLeavesalreadyHaveSolutions)
            addStateToSolutionList()
          else
nextStarter s' = pickStarterWithFewestPanOptions()
propagateOptionToOutgoingLinks(s')
```

The next and even more complex task of the system is the monitoring, observation and reorganisation task during the running production. So far, no multiagent system would have been definitely necessary used although the decentralised local search for an optimal schedule subject to a local objective function combined with the planning problem are easier to handle by a multiagent system instead of a centralised approach. In cases of operational faults during the ongoing production the aggregates themselves have to find a new solution for themselves regarding their local objective functions, other agent affected by these changes just combine these solutions checking whether they are compatible and optimises them in respect to their objective function. This is one main advantage for the use of a multiagent system instead of a centralised approach where this scenario would cause more computational costs.

During the monitoring and execution observation the agents ask continuously for data from the steelwork. The received signals are compared to their calculated schedules. If these signals are inside the calculated time windows, everything is working all right. If the signals are going to run out of time this is detected as early as possible. All results are displayed in so called monitor clients which are accessible from any station of the Supply Chain – in Section 4 this is described in detail. The monitoring window is realised as a Gantt chart depicted in figure 4; it shows the schedules for each aggregate, as well as the schedules of the pans are displayed on a second screen. All errors are marked in the Gantt chart as well as written with additional explanations in textual form.

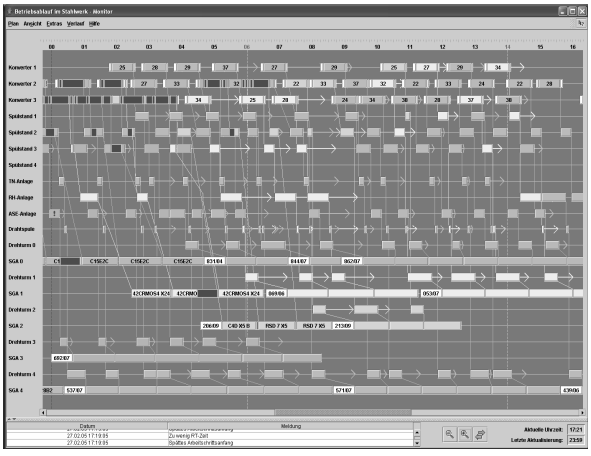

Figure 4: Monitor Client

If an agent recognises that its schedule is running out of the determined time window, he immediately tries to rearrange this. Depending on the severe of the operational fault, the agent can handle this locally or has to contact others which might be affected by his reactions. Following repair strategies are implemented:

a) **Free float checking:** It could be that the aggregate is not used directly after that task and a delay has up to now no bad effect on the following tasks of this charge. Hence, no further actions are necessary.

b) **Internal order changes:** If the first case fails – meaning it has already effects inside this aggregate – this could have a cascading effect over all aggregates inside the steelwork. The agent tries to change the internal schedule. To do so, he has to contact all other agents which could probably be affected by this action. Using the "Contract Net Protocol" [4] he solves this internally with each affected agent.

c) **Alternative aggregate**: It could happen that the aggregate is not able to fulfil that specified task at all under these new circumstances. Now certain other aggregates might do the same task. Using "Simulated Trading" [5] the agent tries to "sell" this task to another which produces less many costs.

d) **Sequence interruption:** In seldom cases a sequence at a pouring aggregate has to be interrupted. Then, a new daily target schedule has to be set up by Saarstahl AG. In a following project this should also be at least supported by the extended system – compare Section 7.

In case a), the system just informs the user that the calculated time windows cannot be satisfied, but that no further actions are needed. The other three cases have much more impact on the complete production inside the steelwork. The reactions of the system on these faults need to be accomplished or even probably modified by the user. Therefore, in these cases a so called Simulation Client is started immediately. This client is realised in another separated window, which is only visible to a view number of persons who might change these settings. The system proposes some ways of how to handle these faults, but the responsible user has to submit them finally. He also has the possibility to start such simulations from himself and compare several potential solutions of how to go on with the production. Not until the user has submitted his decision the actual planning will be changed and the monitoring still shows these faults.

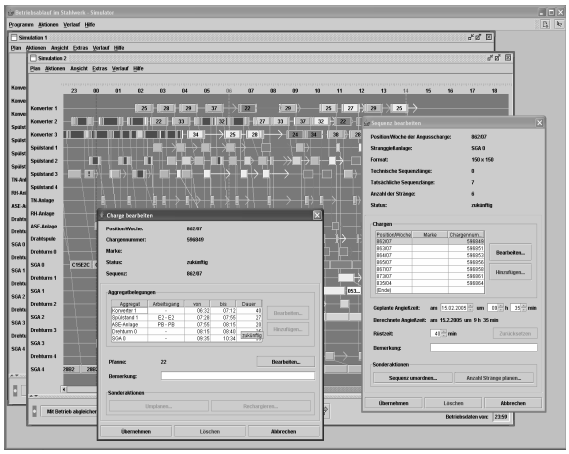

Figure 5: Simulation Client

Figure 5 shows the GUI of the described simulation client. The user has the possibility to change each parameter for the production manually or simply accept the proposed handling of the system. He also has the possibility to start such simulations

only for “what if”-testing or might change parameters the system is not allowed to change but has to handle them.

Every delay of the production of one charge might have influence on the complete production inside the steelwork. Where you might run into trouble, is not realisable at first sight. Each agent – meaning each aggregate and even pan – tries to satisfy its own goals and additional optimisation aspects are also satisfied by specified agents with particular knowledge bases.

Every action is monitored and potential clashes are shown at an early stage of time with additional several repair suggestions from which the user might choose one.

4. SYSTEM ARCHITECTURE

In this section, we give an overview over the agent architecture, the internal architecture of our software system that solves the aforementioned problems. Additionally, we describe the external architecture regarding the embedding of the system into the IT-environment of Saarstahl to ensure the interaction with the other parts of the Supply Chain.

4.1 Agent Architecture

Multiagent systems are particularly well-suited for the scenario described in this paper because they allow a very natural mapping from the entities occurring in the scenario to agents.

InteRRaP [3] is a generic agent architecture for situated agents that integrates reactive behaviour and deliberation. The architecture was designed for agents that exist within multiagent systems and thus some emphasis is on the communication aspect.

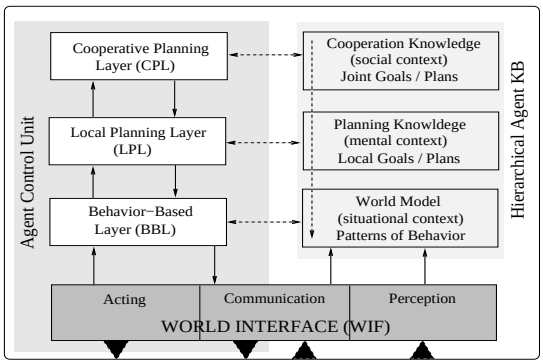

Figure 6: InteRRaP

As depicted in Figure 6, InteRRaP is a layered architecture that consists of three concurrent layers:

- **Behaviour Based Layer (BBL)** This layer implements the reactive behaviour of the agent, i. e. this layer reacts to external requirements without any explicit reasoning, thus it reacts very fast.
- **Local Planning Layer (LPL)** This layer performs the planning process of an individual agent, it is also responsible to monitor the plan execution of the agents current plan.
- **Cooperative Planning Layer (CPL)** This layer is responsible for the coordination with the other agents within in the multiagent system. The coordination with the other agents is achieved with explicit negotiation protocols.

All these layers run concurrently, the intra-agent coordination between the three layers is achieved via the knowledge base. The knowledge base is conceptually divided in three layers (world model, mental model, social model), but each layer has access to the knowledge on every level. The conceptual discrimination, however, allows for a clearer design because most of the information stored in the knowledge base can be associated with a particular layer. The InteRRaP architecture offers a generic framework for agent design that must be instantiated for the particular needs of a concrete scenario.

Concerning our system this architecture is very useful. The Behaviour Based Layer takes care of the signals during the monitoring or new arriving demands from the outside. The Local Planning Layer is responsible for the local optimisations and calculations concerning only this aggregate/agent and finally the Cooperative Planning Layer is used for situations in which interaction with other agents is needed – meaning cooperation inside the steelwork is needed to return to an optimal plan execution. This layer is needed, if operational faults occur which cannot be handled locally and affect the other aggregates as described in the former section.

4.2 Internal Architecture

The internal architecture of the implemented system is a Three-Tier Architecture [7]. Inside the steelwork each aggregate sends data of the task steps it fulfils to a central computer. In certain fixed time-intervals the steelwork computer sends telegrams over a host into the database.

The database as backend of the architecture triggers after a certain amount of information sent by the steelwork over telegrams and sends the data as xml-file to the planner to give a complete snapshot of the steelwork.

The planner – the middle tire – also asks continuously for information of the database, makes a plausibility check and propagates the data to the corresponding agents. The agents have the knowledge to check whether the received real-time data is still compatible to their calculated schedule or if they run into trouble somewhere in the future.

The clients as front end receive the results and visualise them. Here the user has everything available to interact with the system. In specified dialogues they are able to test potential changes before they are updated over the planner into the database. Two of these clients are described in Section 3, other clients like one to enter new daily target schedules are not of much interest here because of their bounded functionality.

The most important objects and their interconnections are depicted in figure 7; the arrows between the objects denote the flow of control or the flow of data, respectively.

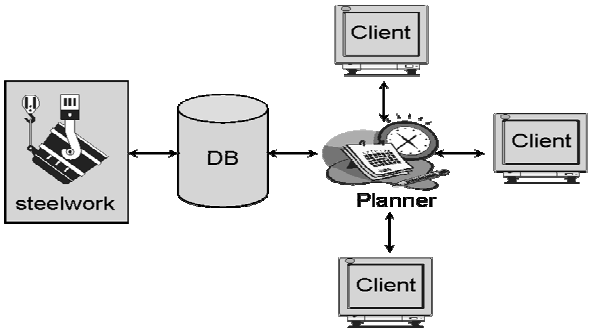

Figure 7: Internal Architecture

4.3 External Architecture

As already mentioned in the introduction, the system is not a standalone system, it is embedded in the IT-environment of Saarstahl with suitable interfaces to interact with the other parts of the complete Supply Chain. Therefore, also an external architecture is needed.

Planner-relevant data like the amount, quality and delivery time of pig iron from the furnace are important. Concerning the following parts in the Supply Chain like the inventories or masticators, the delivery date of the steel or fine adjustment of orders to configure the daily target schedules are relevant to incorporate. The planning system for the production inside the steelwork cannot control these circumstances completely, but the negotiations are possible to support by the agents [8] using Web-Services. Therefore, this external service-oriented architecture [9], [10] has been developed. It provides Web-Service interfaces to the other partners involved, in which these relevant dates can be determined.

We use BPEL [11] to express the real business processes behind these negotiations. Then, we use WSDL [12], [13] to describe the endpoints behind these negotiations, i. e. the involved partners. The messages sent between these partners are realised as SOAP-messages. By using the described service oriented approach as depicted in figure 8, the flexibility is kept and the interfaces are modelled for a further extension.

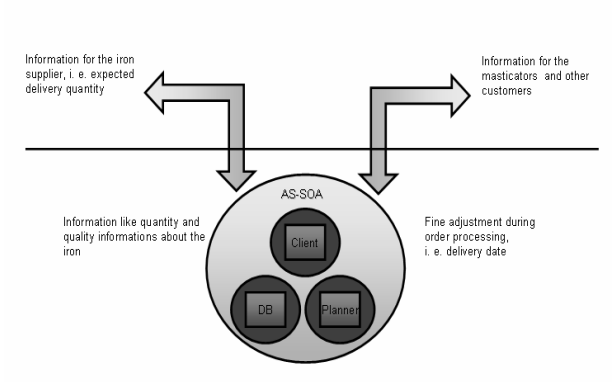

Figure 8: AgentSteel – Service Oriented Architecture

5. CONCLUSIONS AND OUTLOOK

Many systems are not able to monitor these specific requirements and moreover are not able to deal with production control. The agent based approach is able to break down the complexity of these requirements without loosing any flexibility. The agents are able to observe their own schedules and follow their own specific goals.

The cooperation between the Saarstahl AG and the DFKI GmbH will continue. The system is used inside the steelwork and moreover the cooperation will expand. In following projects the calculation of the daily target schedule will be taken into account to make the system even more flexible. The overall goal is to plan and observe by a multiagent system the complete supply chain of Saarstahl AG beginning from the furnaces over the steelwork, the masticators up to the customers. Service Oriented multiagent architectures will be used to handle the interoperability between the companies and increase the flexibility to ensure the competitive position of Saarstahl AG on the world wide market in the steel sector.

6. REFERENCES

[1] G. Weiss, “Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence”, KIT Press, 1999

[2] M. Wooldridge, “An Introduction to Multiagent Systems”, John Whiley & Sons, 2002

[3] J. P. Müller, “The Design of Intelligent Agents: A Layered Approach”, vol. 1177 of Lecture Notes in Artificial Intelligence, Springer-Verlag, 1996

[4] R. Smith, “The contract net protocol: High-level communication and control in a distributed problem solver”, IEEE Transactions on Computers, 1980

[5] A. Bachem, W. Hochstättler and M. Malich, “Simulated Trading: A New Approach For Solving Vehicle Routing Problems”, Tech.Rep. 92.125, Mathematisches Institut der Universität zu Köln, 1992

[6] W. Clark, “The Gantt Chart”, Pitman and Sons, London, 3rd edition, 1952

[7] A. Berson, “Client/Server Architecture”, McGraw-Hill, 1996

[8] N. Jennings, An agent-based approach for building complex software systems, In “Communications of the ACM”, volume 44, pages 35-41, 2001

[9] D. K. Barry, “Web Services and Service-Oriented Architectures”, Morgan Kaufmann, 2003

[10] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris and D. Orchard, “Web Services Architecture”, Working group note, W3C, http://www.w3.org/TR/ws-arch/, 2004

[11] BPEL4WS, Business Process Execution Language for Web Services, http://xml.coverpages.org/bpel4ws.html, 2002

[12] W3C, OWL-S: Semantic Markup for Web Services, Member submission, W3C, http:// www.w3.org/Submission/OWL-S, 2004

[13] W3C, Web Service Addressing – Core, Working Draft, W3C, http://www.w3.org/TR/ws-addr-core, 2004