Designing Agent Based Inter-Organizational Systems: Business and IOS alignment in the Port of Rotterdam

Albert Douma*, Hans Moonen**, Jos van Hillegersberg*,
Bastiaan van de Rakt***, Marco Schutten*

* University of Twente, Enschede, The Netherlands
** RSM Erasmus University, Rotterdam, The Netherlands
*** INITI8 B.V., Rotterdam, The Netherlands

Abstract

Inter-organizational systems (IOS) hold high promises for improving the coordination of activities in logistics business networks. Although several successful IOS have been implemented, success is scarce in situations where different parties like to keep a large degree of autonomy, are hardly prepared to share information and do not have any contractual relationships. In such settings, agent based IOS may be the preferred solution. Agent technology sets another way of looking at system design and architecture. Agent systems come to solutions not through central optimization as most traditional IOS, but as a result of agent communication. However, real-world implementations of agent-based IOS are scarce and little is know regarding how to design such systems to assure successful adoption. In this paper we discuss a real-world agent-based IOS we are designing. The system deals with the problem of barge rotation planning in the Port of Rotterdam – an example in which an agent based IOS seems very promising. This paper reports on how the inter-organizational setting and multi-agent coordination design need to be aligned. We conclude that an agent based coordination mechanism should not only be designed with optimal performance in mind, but should also fit the real-world setting in which strategic and competitive factors dominate.

Keywords Inter-Organizational Systems (IOS), Agent Technology, Supply Chain Management (SCM), Smart Business Networks (SBN), Planning, Scheduling

1. Introduction

Inter-organizational systems (IOS) hold high promises for improving the coordination of logistic operations across organizational boundaries.

Emerging trends such as business process outsourcing, globalization and increasing congestion of transportation infrastructures have increased the need coordinating logistic activities among businesses. Recently, various success stories have been broadcasted to the world by technology vendors, consultants and researchers. Widely cited examples include the IOS established by Dell, Cisco and Zara. However, these cases have in common that the IOS were introduced by dominant parties in the supply chain, who could exercise their power to accelerate the adoption of the IOS by business partners. It is far from trivial how to establish successful IOS in dynamic business networks, where such dominant players are often lacking and different parties usually like to keep a large degree of *autonomy* within the business network. Organizations are often only prepared to *share limited information*, and sometimes do not have *contractual relationships*. In such settings, few success stories of traditional IOS are known. The centralized architecture of traditional IOS targeted at supply chain wide optimization of coordination simply does not align to the inter-organizational setting.

It is our proposition, that in such settings an Agent Based IOS might bring an important revolution.

Agent technology is a relatively new sub domain within computer science, that introduces another way of looking at system design and architecture. Systems are no longer perceived as large monolithic pieces of technology, but rather exist of interacting intelligent agents, all equipped with autonomy, intelligence, certain (proactive and reactive) behaviors and goals, and able to communicate in real-time with other agents. Such systems reach solutions not through central optimization, but rather as a result of agent communication and negotiation.

Agent Based IOS would provide a better fit to the dynamic and decentralized power structure of many of today's business networks. We therefore expect that Agent Based IOS have better chances of being successfully implemented within business networks. However, little is known of how to design Agent Based IOS for optimal alignment to the business setting. Most research on IOS implementation has been focused on adoption factors for traditional IOS. Agent Based IOS have usually not been studied in a real-life context. Davidsson *et al*.(2005), in a survey of the literature on Agent Based Systems for transport logistics, find that "*the vast majority of the approaches surveyed have just reached the level of conceptual proposal or simulation with limited or artificial data*". In this paper we report upon a real-life case in which a system is being designed to deal with the problem of barge rotation planning in the Port of Rotterdam. A dominant actor is lacking in this business network, and earlier attempts to introduce a central IOS to facilitate information sharing have failed. Is an Agent Based

IOS better positioned to be successfully implemented? What design choices can contribute to implementation success?

In the remainder of this paper we address these questions. First, the IOS literature is briefly reviewed. In section 3, we discuss design considerations for agent-based IOS. In section 4 we describe the case setting and describe the problem of barge rotation planning. In Section 5 we present a first design of an agent based IOS for this setting. The paper concludes with a discussion section.

2. Requirements for Inter-Organizational Systems in distributed business networks

Inter-organizational systems are information and communication technology-based systems that transcend legal enterprise boundaries (Kumar *et al*., 1996). The first inter-organizational systems appeared in the late Eighties and the beginning of the Nineties. These systems were mainly based on EDI, had a point-to-point nature, and were rather inflexible. Although the introduction of these systems let us enter the new era of computer-supported inter-organizational working, getting these systems to work in the larger supply chain context was virtually impossible. Although a second generation of IOS technology provided new concepts, such as central planning hubs, it remained hard to establish working systems due to e.g. the different systems and procedures of all the supply chain actors that had to connect to the hub. Two other drawbacks of the centralized system appeared to be: Firstly, it is difficult to encapsulate all the required information that should be ideally needed for decision making – see for example Sridharan (2005), and secondly optimization of such complex models needs a large amount of processing time – hence the fact that an optimization run of a deployed ERP system within a large company could easily take an entire night.

Establishing a centralized planning system that coordinates all the activities of the different (independent) parties involved – such as in the setting of this paper: the coordination of barges and container terminals – is not easy to achieve. Reasons are, among others, that players feel lack of control about their own operations; the central party probably makes decisions that are beneficial for one player but are costly for another one, which is hard to justify. Also, how to share gains obtained by the system is a difficult issue that may hinder smooth implementation and use. Lee *et al.* (2005) and Lu *et al.* (2006) research the critical success factors for inter-organizational system implementation and independently of each other derive to similar conclusions: some of the major influencing factors are a

strong management believes and achieved high-levels of trust between partners.

Ongoing developments in hardware, software, and networks change the way we work, and build systems. Software systems become e.g. more real-time. Independently, the inter-organizational aspect becomes an integral design of systems and supply chain practices (Lee, 2004).

Real time coordination and distributed decision making are inherent properties of agent based systems. In agent based systems players are represented by software agents acting in the best interest of their principals – here a terminal operator or a barge operator. Agent based systems (Wooldridge *et al*., 1995) are software systems, consisting of agents which are small pieces of software that can act *autonomously*, are able to *communicate* with other agents, can *react* to events in their environment and can *anticipate* on events. By means of negotiations agents come to agreements that are binding in the real world. In (distributed) systems the communication aspect is very important, because of all the interactions between many small entities (within an organization or between organizations). In a world where the amount of data is exploding, agents can help in reducing complexity through smart monitoring (Desouza, 2001).

In many inter-organizational settings of the "*pooled interdependency type*" (Kumar *et al.*, 1996), it is our proposition that the different parties involved perceive the following characteristics as very important:

(1) **Autonomy**. Parties want to be in control of their own operations and are not willing to hand this over to a third (trusted) party.

(2) **Limited information sharing**. Parties are not willing to share information that can be used by other parties to improve their competitive position.

(3) **No contractual relationships**. Between parties there can be no contractual relationships. This effects the obligations of players towards each other and can limit the ability of a single party to force others to meet agreements.

Although the parties in a network might have conflicting interests, they depend on each other to operate profitably. Agent-based systems seem to fit well to such a setting; giving each party its own agent, equipped with its own objectives, and intelligence, which interact with the other parties' agents to come to mutually beneficial arrangements. This may overcome some of the most important struggling points of previous IOS's. In addition, Agent-Based IOS promises more benefits using the properties of an agent; *high performance* as it relies only on communication between individual agents; *high flexibility* due to its extendibility, since adding a party does not impact the larger system, it just adds an individual module; *encapsulating knowledge* is one's own responsibility, and not an issue for the

IOS designer/operator. Last, but not least, *control* (of operations) is not given over to a central authority, but handed over to one's own software agent. This agent can share whatever detailed level of *information* it wants to share with the party its communicating with.

3. Design Considerations for Agent-Based IOS

Like the design of any information system, the design of agent based systems goes through several phases (Luck *et al.*, 2004) as Figure 1 illustrates.

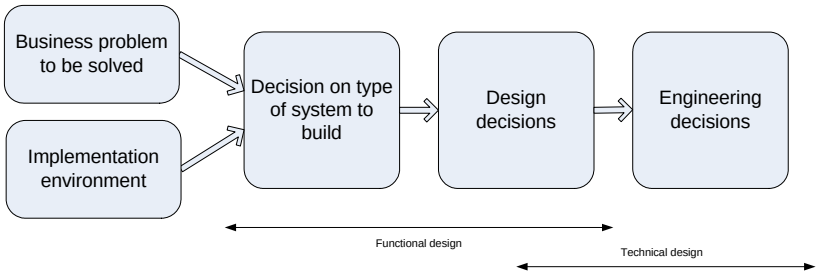

Figure 1 Basic steps in enterprise software designs.

After making a choice for a decentralized Agent-Based IOS, one has to make several design choices as depicted in Figure 2. The business environment leads to a certain **logistical mechanism** to solve the business problem. This can be an approach towards continuous planning, but also a real-time assignment approach which neglects any pre-planning. The **agent societal structure** corresponds to the type of agent environment to establish: are all agents in the system equal, or do we go for a layered hierarchical approach? The **coordination mechanism** is a related and important design decision: how do agents synchronize their plans, agree about decisions, and monitor their performance. **Human involvement** in the system is another important design factor: who does what, how do system and user interact, and what is autonomously performed (by the system's agents) and what tasks are handled by human actors? **Agent intelligence** is also important to take into account: can the system learn, and analyze data, or should it just do its tasks as built-in at design time. Finally, a choice for a **technology** platform is needed: which is related to the choices made in previous steps, but is also influenced by the experience of the designers and programmers involved.

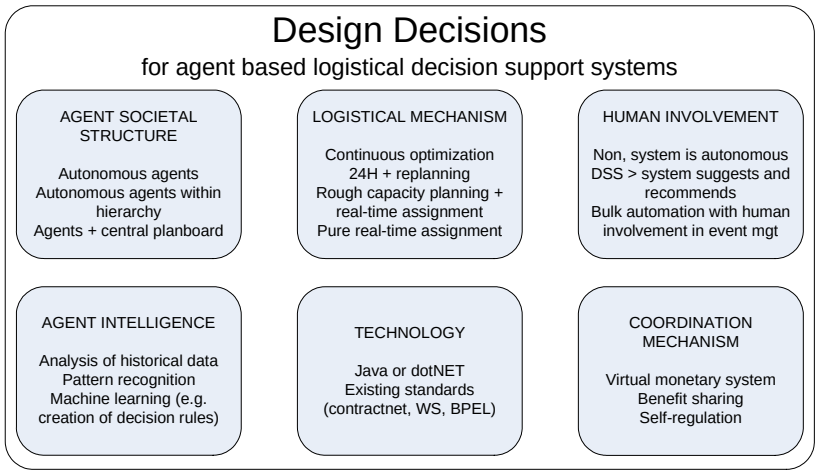

Figure 2 An example of some of the specific design decisions to be made for agent based system design

4. The IOS setting: Barge rotation planning in the Rotterdam Port

One of the results of the ongoing globalization is a large increase in good flows around the world. The Port of Rotterdam handled 5.6 million containers in 2005. Most of these containers have to be shipped to the hinterland by truck, train or barge. In this paper we specifically look at the container barge sector, and the problems that exist in that domain to align barge visits with terminal capacity in such a manner that terminals can achieve high quay utilization and barges can leave the port according to their sailing schedule.

This coordination problem is rather complex. Every day about 75 barges visit the port, and each barge has to visit on average eight terminals. In total there are more than 35 terminals in the port. Many of these also handle sea going vessels and some are closed during the night, which limits the possibilities for barges to plan a rotation. Moreover, there is no contractual relationship between terminals and barges, which means that they cannot force each other to provide predefined service levels. Currently, coordination is done by means of telephone, fax or e-mail. However, this takes too much time and due to changes and disturbances it is often not possible to execute a certain rotation or quay plan as it was planned. Traditional centralized information systems did never gain ground, due to various reasons – competitive issues being one of the most important ones.

Therefore we proposed a different approach: supporting this inter-organizational planning and scheduling problem through a decentralized agent-based approach. The proposal is funded as part of a nationally funded program; Transumo: Transition to Sustainable Logistics.

The requirements we gathered for the project came from various earlier studies into the barge-terminal coordination problem in the port of Rotterdam (Melis *et al.*, 2003; Connekt, 2003; Schut *et al.*, 2004; Moonen *et al.*, 2005). Dedicated interviews and workshops with industry representatives were another source to gather industry requirements, and gain insight in the differences between barge-operators and terminal-operators. Furthermore, as part of this project work we interviewed executives and planners in the port to identify key performance indicators for the different parties involved (Van Groningen, 2006): this gave us interesting insights in how responsibilities are divided over the chain, and how different parties react upon each other. These elements are important for an agent-based control system which is implemented in an inter-organizational setting.

4.1 Business problem

Current manual planning of barge-rotations is complex: many factors complicate the planning, among others:

- *Physical layout of the port*. The terminals are spread over the port, mainly in three different clusters: city, Botlek and Maasvlakte. Sailing from one end to the other takes a barge around three hours. This impacts the rotation planning.
- *Physical layout of container terminals*. Sea going vessels are handled at the same quay, and terminals generally also handle road and rail activities. Capacities and layout are restrictive for the quay planning.
- *Existing systems*. Many systems are in use throughout the port among different parties. With some of these systems interfacing is needed.
- *Type of journeys*. We distinguish three different types of journeys: Rhine, Antwerp and inland domestic journeys. The type of journey influences the rotation of a barge. E.g. Antwerp traffic visits only a limited amount of terminals, but tends to have large quantities. Rhine in contrary results in many visits, and inland domestic e.g. is served with smaller barges.
- *Three important decision moments throughout time*: 24h, 4h and 0h before execution. The current practice shows a division in three important decision moments. 24h in advance (i.e. every day before a specific time) terminals have to announce the amount of shared labor they need for the next day. 4h prior to processing a barge has to announce which containers it has to load, to let the terminal stack the containers timely

at the quay. 0h the terminal makes some operational decisions, like which team is processing a specific barge.
- *Late orders / plan disturbances*: unexpected events such as late orders and breakdowns have a disturbing effect on planning and operations.

4.2 Implementation environment

Important design considerations are related to the implementation environment; several issues play a role:

- *Many different players with conflicting objectives*. In the system different players are involved, like terminal operators and barge operators, which have different objectives. This means that the system must meet the objectives of both groups of players, which are conflicting to a certain extent.
- *Highly competitive environment*. Barge operators compete with eachother and do not want to give their competitors a inside view on their operations; the same holds for terminal operators.
- *Strategic behavior*: Currently strategic behavior by the actors worsens the way the current systems works. Especially barge operators try to increase their opportunity to be processed timely by exaggerating the time they need for processing or just ask for a limited amount of time, knowing that once they are processed the terminal will not send them away. Data analysis of execution data from PortInfolink's BargePlanning (a platform that currently facilitates EDI message exchange between a limited group of barge operators and terminals) and interviews revealed that strategic behavior takes place to a substantial extent.
- *Gain-sharing*. Gain sharing and therewith proper performance measurement turns out to be very important, in order to get the system implemented and adopted. Gains should be measured for every player individually and for the system as a whole. If the system does not meet the expectations of a single player, this player has an incentive to quit, which is not desirable.
- *Yearly growth*. The container market has been growing yearly over the past decade with double digits, and the same is foreseen for the near future.
- *It is hard to establish a trusted party that coordinates all operations.* Barge and terminal operators will not quickly accept an authority that is coordinating everyone's actions, due to the fact that they want to stay autonomous and in control of their own operations. Moreover, registration of e.g. performances will be hard which complicates the task of a possible authority of trusted party.

5. Design of the Agent Based IOS

In this section we present a design of the agent based IOS mainly focusing on the logistical design mechanisms required (as shown in Figure 2). The question we consider is how such a system can provide an efficient coordination mechanism taking the unique requirements of the business context into account. We therefore have to define agents, design a communication mechanism and to equip agents with some form of intelligence. Based on these results we can evaluate whether and when it is beneficial for terminal and barge operators to participate in the system and how the system performs over time. We first describe which entities are modeled in the agent system and how the communication among the agents can take place.

5.1 The Agents

To define agents we have chosen for a physical decomposition of the system since this comes closest to the current situation. This means that every barge operator and every terminal operator is equipped with an agent operating in their best interest.

To keep the design simple, we chose to establish a system with only two different types of agents: barge operator agents, and terminal operator agents. We assume here that all barge operator agents are equal and the same holds for all terminal operators agents. So, every barge operator agent would make the same decision if it would be in the same situation as an arbitrary other barge operator agent. We focus on the question which coordination mechanism is appropriate for the problem we consider.

Before designing a coordination mechanism we first have to define the goals of each agent. The goals of both the terminal and barge operators we derived from the interviews mentioned earlier (Van Groningen, 2006). To simplify the analysis we first focus on the main goal of both the terminal and the barge operator, leaving apart all the secondary objectives. The main goal of the terminal operator is its utilization degree. The main goal of the barge operator is to leave the port within the time window that is set in its sailing schedule. Despite this simplification, reality is still complicated as a terminal operator decides only once a day on the amount of capacity it is going to operate the next day. For simplicity, we first assume that the capacity of a terminal is constant.

One might wonder whether the simplified problem setting (with a single objective for all terminal and barge operators) is still realistic. Obviously the reality is far more complicated than the model we now consider. However, we are firstly mainly interested in the question whether and to which extent a distributed approach for this business environment can lead to optimization of the terminal and barge performance.

5.2 Coordination Mechanisms

5.2.1 General communication mechanism

The communication mechanism is very important for the way agent intelligence is developed. Agents can communicate directly (negotiation), or indirectly by means of the contract net protocol (Luck *et al*., 2004). However, the latter requires a kind of (virtual) currency that is exchanged between parties, which does not match to the current situation where there are no contractual relationships between terminals and barges. We propose a direct communication mechanism (negotiation) which mirrors daily practice and is probably easier accepted by barge and terminal operators. Direct communication means that every barge operator contacts all terminal operators it has to visit during its rotation about convenient times to load and unload. Based on the response of every terminal operator it decides when it is going to visit every terminal and in which sequence. This is actually the way the manual system works at the moment.

To illustrate the proposed agent model we depicted one barge operator contacting several terminal operators to come to an agreement about the time it can be handled – see Figure 3.

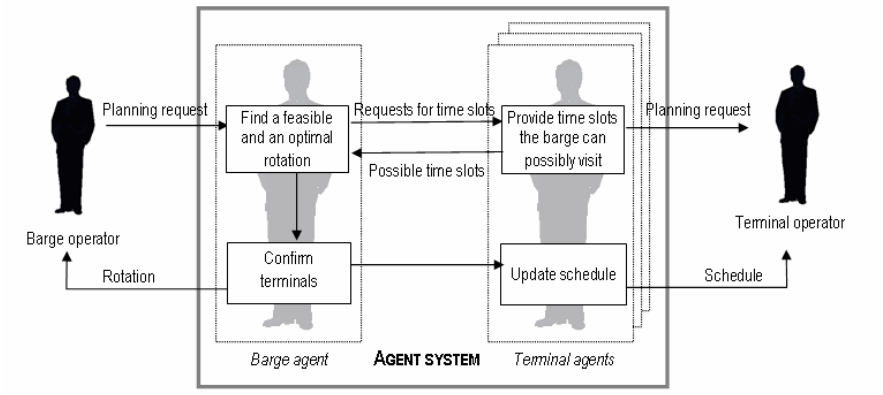

Figure 3 Illustration of the working of the agent model

It can be necessary for a barge to cancel terminal visits if it becomes clear during operation that appointments cannot be met due to disturbances. This is especially important when terminals refuse to process a barge if it arrives too late. If a terminal refuses to process a barge, this barge has to make new appointments. This probably means a delayed rotation since it is likely that a terminal has no free capacity in the first couple

of hours. A barge would therefore try to make robust rotation plans which are less sensitive for disruptions like delayed processing or sailing times.

5.2.2 Analysis of possible implementations of the agent-communication mechanism

The choice for a suitable communication mechanism is not trivial; several mechanisms can be considered. We first discuss these alternative mechanisms and then propose the mechanism that seems most promising in more detail.

A first option is just automating the way communication is done in practice. In this case, the Barge Operator Agent (BOA) (on behalf of the barge) contacts all the relevant terminals and asks the corresponding Terminal Operator Agent (TOA) whether the barge can be processed at a specific time. If each TOA agrees on the time proposed by the BOA, the rotation can be fixed. However, if only one TOA disagrees, the BOA has to change the rotation, i.e., the sequence in which the terminals are visited or the times at which the terminals are visited, and propose every TOA another time. This process continues until the BOA has created a rotation that is feasible, i.e., every terminal expects the barge at the time that is determined in the rotation. However, it can take a lot of communication before a BOA has an agreement with all the TOA agents. Although messages can be exchanged fast, every request has to be processed by both the Barge Operator Agent (BOA) and the Terminal Operator Agent (TOA) which takes time. Moreover, for a BOA it becomes more difficult to optimize its rotation, since it does not know what alternative time slots are for visiting a terminal. The TOA conversely has only limited possibilities to plan a barge visit at the time it prefers.

The second possibility would be, just to plan a rotation and then ask terminals successively for the first opportunity to be processed. In that case a rotation is only based on travel times. However, the time a barge is in the port does not only depend on travel times but also on waiting times at every terminal. A BOA is willing to accept a bit more traveling if this leads to a reduction of the rotation time and as long as it is profitable. In this way of communicating a BOA is not able to weigh travel and waiting times such that its rotation time is minimized. Although the TOA has more opportunities to process a barge at the preferred time, the result can be very negative for the barge.

The third possibility is not to ask for one feasible time slot, but to ask for a number of time slots instead. This means that a TOA provides a BOA all the time slots available for this barge. Based on the provided time slots a BOA optimizes its rotation and confirms the times it likes to visit the terminal. However, providing time slots has some important limitations.

Firstly, it can happen that a barge cannot find a feasible rotation or deals with incomplete information (not every terminal has provided time slots for some reason) which complicates the construction of a rotation. Secondly, it is more difficult to let the barge apply for a preferred time slot or a preferred time within a time slot.

5.2.3 A two stage approach using waiting times

To overcome these limitations we propose an alternative two stage approach based on waiting times. In the first stage a BOA asks a terminal for the expected waiting times during a day and the terminal replies to this BOA with a waiting profile. This waiting profile – see e.g. Figure 4 – shows the expected waiting time during the day and can be customized for a specific BOA based on its characteristics or its reputation. The expected waiting time is a maximum waiting time as well. This is a service to barges so that they have more certainty about the time their processing is finished. The barge has to make sure to be at the terminal at the time it promises to be. If it does not arrive in time, its reservation is cancelled and a new appointment has to be made. Waiting times have an additional advantage since they give possibilities for a Terminal Operator Agent (TOA) to keep some slack in its schedule to cope with uncertainties. In fact, exchanging waiting times is more general than exchanging time slots since the latter can be derived from the former.

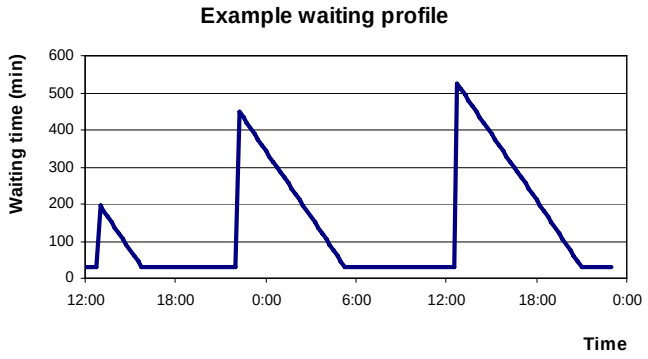

Figure 4 Example of a waiting profile sent to a BOA by a TOA

The second stage consists of constructing a rotation and making appointments with the terminals – see Figure 5. The rotation is constructed based on the waiting profiles and the expected waiting times. A BOA aims to find a rotation that minimizes the sum of the expected waiting, handling and sailing times. Once a BOA has determined the best rotation, it an-

nounces the time it expects to arrive at every terminal and receives a confirmation. A barge obliges itself to be at the terminal at the announced time. If not, it has to make a new appointment and its reputation at this terminal can get hurt. This reputation can be used in future models to adapt waiting profiles and to force barges to incorporate enough slack in their rotation to keep it feasible. This aspect results in a kind of self-regulation in the system, i.e., barges cannot deviate too much from their appointments although this behavior is not contractually enforced.

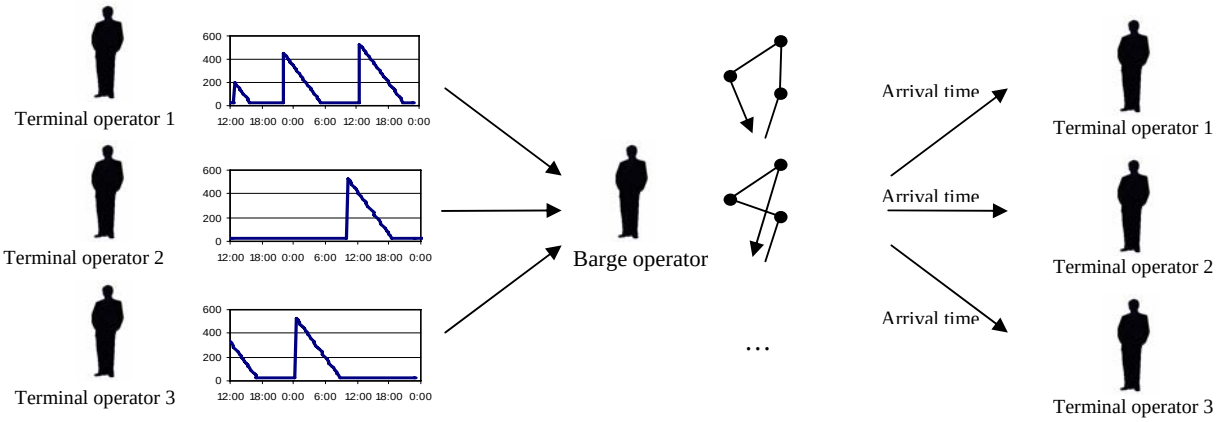

Figure 5 Illustration of the inner mechanism of the agent system, utilizing the waiting profiles.

5.3 From Static to Real-Time Planning

Another important decision before developing an agent system is whether or not plans are made in real-time or at fixed moments in time (batch mode) – e.g. 24h in advance. In the latter case the request of all barges have to be submitted to the system at e.g. 10:00 every day to make a plan for the next day. This has some disadvantages however, because plans need to be revised probably when disruptions occur and rotations become infeasible. Real-time planning, on the contrary, is more flexible since plans are constructed and adapted during the day. In this paper we implement a 'near' real-time planning system, or to be more specific, a dynamic-deterministic planning system. Dynamic here means that barge requests arrive during the day, which means that information is revealed over time.

Static means that information about *all* barges is known at the start of the planning process. Deterministic means that there is no uncertainty in sailing times and handling times, and that there are no disturbances during operations.

Although in the real-world setting, disturbances will occur, we first focus in our research on the transition to real-time planning. Our aim is to first establish deterministic algorithms and evaluate their performance in different scenarios, determined by the utilization of the network, the balance of containers flows, closing times of terminals et cetera. Once these are established, in future research we will also study the effects of disturbances on the robustness of the planning. Types of robustness that a stochastic planning methods needs to deal with will be discussed briefly in the future research section.

5.4 Experimental Design

In this section we set our experiments. First we show results from experiments done in a workshop with logistics planners from barge- and terminal operators. The result clearly illustrates the problems in the current situation. Then, we describe two dynamic algorithms and one static algorithm. The latter is used to compare the performance of the dynamic algorithms.

5.4.1 Current situation

The way barge rotations and terminal capacity are currently aligned does not only lead to inefficiencies but even to infeasible rotations or quay plans. This was clearly illustrated in a workshop with barge and terminal operators held in September 2004. More details about the workshop and the results can be found in Moonen *et al.* (2005). In the workshop barge and terminal operators had to align barge rotations and terminal capacity the way they are used to do it. Afterwards the results were compared with an basic multi-agent implementation of the system. The goal of the workshop was to clearly show barge and terminal operators the problems in the current situation and the great potential of an agent based solution. The focus was mainly on creating feasible plans, which would mean a big improvement compared to the current situation.

The results of the workshop indeed indicated that, although barge and terminal operators tried their best, they could not even come up with feasible plans. Some barges had to visit two or three terminals at the same time. The agent-based implementation, on the contrary, was able to provide feasible plans in reasonable time. The impact of infeasible plans is severe, since infeasibilities lead to unused capacity and a rapid propagation of this

inefficiency to other terminals. Once rotations or quay plans become infeasible, barge and terminal operators easier decide to deviate from their plans, thus worsening the situation.

5.4.2 Setup of experiments

To illustrate the performance of an agent based system we show results based on the dataset used in the workshop. This gives us the opportunity to make a comparison with the "real-life situation". Comparison based on actual data derived from the real-world is hard for several reasons. Firstly, we need an accurate and extensive dataset reflecting daily practice and can provide insight in the working of the system would had have been implemented. Secondly, it is not possible at the moment to collect for one (let alone for more than one) case all the relevant data necessary for a comparison. Thirdly, making a comparison between the plans made in practice and the plans that are made by the agent system is not possible since most of the plans made in practice are infeasible.

We simulate two different dynamic-deterministic algorithms. In the first algorithm barges optimize their rotations based solely on travel times. After they have determined the shortest path to travel to the port, they ask the terminals successively for the first possible processing time. So, barges do not consider waiting times to optimize their rotation. We refer to this algorithm as the 'first dynamic model'.

The second algorithm is based on the two stage approach described in the previous section. Barges arrive one by one at the entrance of the port (Brienenoord) and plan their rotation based on travel times and expected waiting times. This algorithm we refer as the 'second dynamic model'.

Both dynamic algorithms, which are described more extensively in the next sections, we compare with a traditional (not agent-based) algorithm which is static and deterministic, i.e., it knows everything in advance with certainty. The static-deterministic algorithm plans all barge rotations with the objective to minimize the maximum lateness, i.e., to minimize the delay of the barge that is delayed the most. Delay can also be negative, when all barges can leave the port in time. In the next section we describe this algorithm. We refer to this algorithm as the static benchmark.

To compare the performance of the algorithms, we use the planning data from the workshop. The setting used in the workshop was as follows. In total 22 rotations had to be planned along eight different terminals in a period of 24 hours. A rotation consists of at least four and at maximum eight terminal visits. Sailing times are calculated based on an average sailing speed of 15 km/h and the real sailing distance between terminals. Handling times are equal to the average time to handle a container in practice. For every rotation the expected time of arrival, the expected time of departure,

the number of terminal visits and the number of containers to load and unload per terminal are given.

5.4.3 First dynamic model

The first dynamic model is rather simple and works as follows. Every Barge Operator Agent (BOA) determines a sequence of terminal visits that minimizes the travel time through the port. We use a nearest neighbor heuristic to construct and k-opt to optimize the sequence (for further reading we refer to e.g. Michalewicz (2004)). The nearest neighbor heuristic constructs a rotation by sequentially adding the terminal that is closest to the current route. The resulting sequence is not necessarily optimal. To improve the sequence, a k-opt heuristic is applied which swaps k terminals in the sequence and evaluates whether this leads to an improvement. If an improvement is found, the new sequence is adopted and the heuristic continues to find improvements by swapping k terminals again. If no improvements are found, k other terminals are swapped et cetera. This is repeated several times till no improvement is made for a certain time.

Based on the sequence that results from the described procedure, the barge operator calculates the time it can arrive at the first terminal and asks the terminal whether it can visit at that time. The Terminal Operator Agent (TOA) returns the first possible time after the requested time. Based on this time the BOA calculates the time it can arrive at the second terminal and asks whether it can be processed at that time. The TOA of the second terminal again returns the first possible time. This process continues until all terminals are planned. It can happen that the rotation is longer than the planned stay in the port

5.4.4 Second dynamic model

The second dynamic model is based on the two-stage approach described in Section 5.2.3. In the first stage terminals provide a barge a waiting profile, indicating the maximum waiting time at every moment of the day. The waiting profile is calculated based on available time slots. To determine available time slots the terminal has to take planned sea vessels, barges and closing times into account. However, the calculation of available time slots is a bit more complicated, since all the planned barges agreed on a maximum waiting time after their expected arrival time. Consider the request of one specific barge. To find all the time slots in the current schedule a terminal operator has to evaluate for every point at which the barge can be inserted in, the current schedule what the earliest and latest possible starting time is. It can calculate this by planning all barges scheduled before the time slot we consider at their earliest starting time,

and all barges after the time slot we consider at their latest starting time. The terminal operator evaluates this for all possible time slots, i.e., for all possible points at which the barge can be inserted in the current schedule.

Once a terminal has determined all available time slots it calculates for discrete moments in time (e.g. every 15 minutes) the expected waiting time for the next available time slot. After calculating all the maximum waiting times for the whole planning horizon, the terminal increases all the waiting times with a certain amount of 'minimal waiting time'. This means that a barge calling for processing within an available time slot, still faces some waiting time. This minimal waiting time gives the terminal the opportunity to change the starting time of the barge to be more flexible toward future barges.

The rotation of the barge is calculated using a traveling salesman heuristic, minimizing the sum of the sailing, handling and waiting times. We use a nearest neighbor heuristic to create and apply k-opt to improve the rotation.

5.5 Static Benchmark

The static algorithm, we use to compare the dynamic algorithm, uses the Resource Constrained Project Scheduling Problem (RCPSP; see, for example, Demeulemeester and Herroelen (1992)) as a base model. In the classical RCPSP, a single project, consisting of a number of activities, has to be scheduled. To process the activities, a number of resources are available. Each resource consists of a number of parallel processors. Each activity requires during its processing a number of units of each resources (for example, 2 units from Resource 1 and 5 units of Resource 2). Between activities, precedence relations exist. An activity cannot start before all its predecessors are finished.

For solving the RCPSP, a graph representation is widely used in the literature. This graph representation is known as an 'activity-on-node network'. See Figure 6 for the activity-on-node network for a simple project, consisting of three activities, which all have a node in the graph. There are precedence relations between Activity 1 and Activity 3, and between Activity 2 and Activity 3. This means that Activity 3 can only start when both Activity 1 and Activity 2 are finished. Besides the nodes representing an activity, there are two dummy nodes s and t. Dummy node s represents the project start and dummy node t represents the project end. If we give each node a weight that is equal to the required processing time, then the length of a longest path from s to a node i is equal to the earliest possible starting time of Activity i. The length of a longest path from s to t is then equal to the earliest possible completion time for the project.

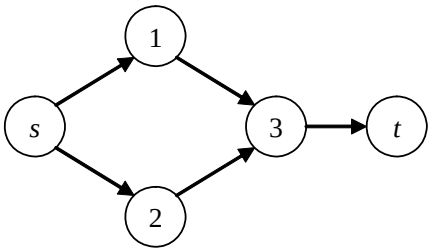

Figure 6 Example of an activity-on-node network

We represent the terminals by resources in the RCPSP. The number of processors for each resource is equal to the number of quays of the associated terminal. In this way, we model that a terminal can handle two barges at the same time if it has two quays. Each terminal visit of a barge is represented as an activity. The processing time of an activity is equal to the load and unload time of the associated terminal visit.

We need to change the properties of the activity-on-node network in a number of ways to deal with characteristics that are not modeled in the basic RCPSP. For example, we give an arc from s to a node a weight that is equal to the arrival time of the associated barge in the port. In this way, the earliest possible starting time of this activity is at least equal to this arrival time. Moreover, we give an arc between two nodes (representing terminal visits of the same barge) a weight that is equal to the sailing time from the first terminal to the second.

For finding a barge rotation, the first terminal to be visited is fixed, whereas the remaining terminals to be visited can be visited in any order. To handle this in our model, we introduce a fictive resource for each barge with only one processor. Each activity associated with a barge requires, beside the terminal resource, this fictive resource. Since this fictive resource has only one processor, the activities associated with the terminal visits of a barge, have to be sequenced. In other words, in a feasible schedule, no two activities from a barge can be processed in parallel.

Finally by having a resource profile, in which the number of available processors for a resource may vary over time, we are able to deal with terminals that are not available at certain moments in time. This is required for terminals that are not open 24 hours a day and for dealing with reservations for sea vessels, which have priority in planning over the barges.

To solve the resulting problems, we used an algorithm that is based on the adaptive search algorithm by Kolish and Drexl (1996). In this algorithm, a (large) number of schedules is generated based on randomized priority rule scheduling. For the basic RCPSP, this algorithm finds schedules that are close to optimal very fast. We adapt this algorithm, for exam-

ple to deal with the resource profiles, which are not present in the basic RCPSP.

5.6 Numerical Results

Table 1 presents the results from a comparison performance of the algorithms using the workshop data. It is clear that a dynamic model based on waiting times and travel times in this case results in a much better performance, than the dynamic model which only uses travel times to plan a rotation. The solution provided by the static benchmark is optimal in terms of the objective and was obtained within one minute of processing time.

In Table 1 we can see that increasing the minimum waiting time to one hour (i.e. minimum amount of maximum waiting time) has a negative effect of the average and maximum lateness of the barges. This is because every barge calculates the worst case, i.e., the maximum waiting time at every terminal. It is quite well possible that the barge indeed has to wait the maximum waiting time if the terminal plans another barge before him (which it cannot know). This means that barges plan their rotation such that all appointments with terminals can surely be met. This leads directly to an increase of the length of the rotation.

If waiting times are high at terminals, it can perhaps become beneficial for barges not use the worst case (the maximum waiting), but a fraction of (or the expected) waiting time. If a barge notices during operations that appointments with terminals later in its schedule cannot be met, it cancels these appointments and makes new ones. However, terminals are probably highly utilized in the first couple of hours, which can mean the barge has to wait several hours for a new time slot. This can mean that revising a rotation during operations can be less attractive than using the worst case scenario, i.e., the maximum waiting times at terminals.

Model		**Maximum lateness**	**Average lateness**	**No barges delayed**
Static benchmark		-24 min.	-480 min.	0
Dynamic model 1		696 min.	-278 min.	8
Dynamic model 2				
Min waiting time equal to…	0 min.	140 min.	-473 min.	2
	30 min.	35 min.	-420 min.	1
	60 min.	140 min.	-362 min.	3

Table 1 Results from three different models

One can see that in this case it is better to have a minimum waiting time greater than zero. This is what we would expect as well. The reason is that terminals have more freedom to use their idle time (they can move the starting times of barges). This is beneficial for the terminal and for the barge as well, since it becomes more likely that a barge can be processed earlier. A minimum waiting time of zero means that a barge can have a maximum waiting time of zero when it calls for processing in an available time window. This can be negative however for the terminal, since the possibilities to fill remaining idle time become very limited which has a negative effect for future barges as well (a terminal is not able to increase its gaps a little bit). We expect that the amount of minimum waiting time depends on the utilization of the terminals. If terminals are highly utilized it is probably better to have more minimum waiting time, whereas in less occupied times a minimum waiting time of e.g. zero can be sufficient.

6. Discussion

In this paper we discussed the design of inter-organizational systems (IOS) for the coordination of logistic operations among different actors in a business network. We discussed this using a case in the port of Rotterdam. We consider a business network where a central platform which coordinates the activities of all companies involved cannot be established for several reasons. A first reason is that all companies participating in the business network want to stay autonomous, i.e., in control of their own operations. Secondly, companies have competitive relationships and are very reluctant to share information that can be beneficial to other companies. Finally, there are no contractual relationships among the companies, meaning that certain performance cannot be contractually enforced.

We state that an Agent-Based IOS in these kinds of networks can provide a valuable perspective over the more traditional centralized architectures. Agent-Based IOS are decentralized, low structured and able to adapt quickly to a changing environment which are properties of the business network as well. Moreover, every agent can operate autonomously, in the interest of a specific company and can encapsulate knowledge. An Agent-Based IOS can mirror to a large extent the way a network is organized in practice and can therefore support optimization of processes where this was not possible with traditional centralized IOS.

However, the design of these systems is complicated due to the constraints from the business and implementation environment. Besides logis-

tical aspects we have to think about several other design choices and aspects as well, like the way self-regulation of the system can be realized using certain communication mechanisms. Other issues are the agent societal structure, the human involvement, and agent intelligence – see Figure 2 for an overview. To develop a coherent Agent-Based IOS all these issues have to be addressed in relation to each other, which makes the design complicated.

In this paper, we have focused on the coordination processes and agent design choices. To show that an Agent-Based IOS can support the coordination of logistical processes we designed an agent structure and coordination mechanisms and showed by means of experiments that an agent-based system can perform close to optimal where practitioners were not even able to find a feasible solution using their current coordination methods.

The case of the port of Rotterdam clearly illustrates that the choice for a coordination mechanism in an IOS is closely related to the business and implementation environment. A coherent design is important because it impacts the acceptance of the system.

This paper presents the current status of our research into the design and implementation of a multi-agent system for barge-terminal coordination in the port of Rotterdam. The project aims at delivering a prototype system that can be used to assess both technical and organizational feasibility. For such an assessment, the prototype needs to mimic reality and its logistical and technical performance should be easily translatable to the real-world. Only on the basis of such a prototype, stakeholders in the port can take a well founded decision whether and how to implement an agent based IOS.

The system that was discussed here still has several limitations. The results obtained by the algorithms are derived using the dataset of the workshop (Moonen *et al*., 2005), so we can not yet draw generic conclusions about the results we presented. More extensive simulations are needed to evaluate the performance of the proposed communication mechanism, the design of more sophisticated barge and terminal algorithms and the extension to more realistic settings like introducing disturbances or refining the objective functions of the agents. Simulation with real-world data is an instrument that we perceive as very valuable in this respect.

An important step in the design of a more realistic is the transition from a deterministic to a stochastic model. This step has both consequences for the algorithms of barge operators and the terminal operators as well as for the performance indicators used for by system as a whole, and by the actors (agents) in particular. The performance indicators change, because not only the expected utilization or rotation time is important, but also the probability that a rotation or a quay schedule can be executed considering disturbances that possibly happen. This means that besides performance of

a schedule also the robustness of a schedule is important. A robust schedule is usually defined as a schedule that remains of high quality when the environment deviates from what was initially projected (Leus, 2003). In a robust schedule, some uncertainty in the execution is taken into account. Further research is needed to develop algorithms that can deal with uncertainty caused by disturbances like delayed sea vessels, delayed barge arrivals, administrative reasons or broken equipment.

Another issue that is critical to real world implementation is the design of the human-multi-agent interface of the system. Wooldridge (2005) mentions that it can be difficult for the principal to accept the decision of its agent, if it is not clear how the agent made the decision or when the principal is uncertain about the extent to which his interests are represented by the agent. In future experiments we plan to evaluate various human-multi-agent interfaces to find out how professionals prefer to interact with the system.

Acknowledgements *This work is partly performed within the DIPLOMA research project, which is one of the current research projects within the TRANSUMO research program www.transumo.nl. We are very grateful for the contributions of Niels Lang, Laurent van Groningen, Ian Miller and Peter Schuur.*

References

Banker, R. D. and R. J. Kauffman (2004). "The evolution of research on information systems: A fiftieth-year survey of the literature in Management Science." Management Science 50(3): 281-298.

Carr, N. G. (2005). "The End of Corporate Computing." MIT Sloan Management Review 46(3): 67.

Connekt (2003). "Eindrapport APPROACH" Participants: Gemeentelijk Havenbedrijf Rotterdam, INITI8, Illyan and the Vrije Universiteit Amsterdam; Connekt, Delft, October 2003.

Davidsson, P., L. Henesey, L. Ramstedt, J. Törnquist, F. Wernstedt (2005). An Analysis of Agent-Based Approaches to Transport Logistics *Transportation Research Part C: Emerging Technologies*, Vol. 13(4), pp. 255-271, Elsevier, 2005.

Demeulemeester, E.L. and W.S. Herroelen (1992). "A branch-and-bound procedure for the multiple resource-constrained project scheduling problem." Management Science, pp. 1803-1818, 1992.

Desouza, K. C. (2001). "Intelligent agents for competitive intelligence: Survey of applications." Competitive Intelligence Review 12(4): 57.

Friedman, T. (2005). „The world is flat – A Brief history of the twenty-first century." Farrar, Straus and Giroux, 2005

Jensen, M. (2001). „Robust and Flexible Scheduling with Evolutionary Computation", PhD thesis, Department of Computer Science, University of Aarhus. 2001.

Kolisch, R. and A. Drexl (1996). "Adaptive search for solving hard project scheduling problems." Naval Research Logistics, pp. 23-40, 1996.

Kumar, K. and H. G. v. Dissel (1996). "Sustainable collaboration: Managing conflict and cooperation in interorganizational systems." MIS Quarterly 20(3): 279-300.

Luck, M, R. Ashri, M. D'Inverno (2004), „Agent-based software development", Artech House, London, ISBN 1-58053-605-0

Lee, H. L. (2004). "The Triple-A Supply Chain." Harvard Business Review (October 2004): 102-112.

Lee, G. G., H. F. Lin, et al. (2005). "Influence of environmental and organizational factors on the success of internet-based interorganizationaI systems planning." Internet Research 15(5): 527-543.

Leus, R., (2003). The generation of stable project plans, Complexity and exact algorithms, Ph.D. thesis, University of Leuven.

Lu, X. H., L. H. Huang, et al. (2006). "Critical success factors of inter-organizational information systems- A case study of Cisco and Xiao Tong in China." Information & Management 43(3): 395-408.

Melis, M.; Miller, I. Kentrop, M; Eck, B. van; Leenaarts, M.; Schut, M.; Treur, J. (2003); "Distributed Rotation Planning for Container Barges in the Port of Rotterdam" In: Verduijn, T. and Loo, B. van de (eds.): Intelli-gent Logistics Concepts. Eburon Publishers, 2003. ISBN: 9051669739. pp 101 – 116.

Michalewicz, Z (2004). How to solve it: Modern Heuristics, Springer-Verlag Berlin and Heidelberg GmbH & Co. K (Aug 2004).

Moonen, H., Rakt, B. van de; Miller, I.; Nunen, J. Van; Hillegersberg, J. Van (2005); "Agent Technology supports Inter-Organizational Planning in the Port"; ERIM Report Series Research in Management, ERS-2005-027-LIS

Moonen, H.; Lang, N.; Nunen, J. Van; Velde, S. Van de (2006). "Creating Virtual Capacity - Utilizing real-time information to boost the performance and reliability of container supply chains", Proceedings of the Transumo Seminar on Infrastructure Reliability, June 2006

Nissen, M.E., Sengupta, K. (2006), "Incorporating software agents into supply chains: Experimental investigation with a procurement task.", MIS Quarterly, Vol. 30, No 1, 2006

Schut, M.; Kentrop, M; Leenaarts, M.; Melis, M.; Miller, I. (2004); "APPROACH: Decentralised Rotation Planning for Container

Barges". Proceedings of the 16th European Conference on Artificial Intelligence, Prestigious Applications Intelligent Systems Conference (PAIS), Valencia, August 2004.

Sridharan, U. V., W. R. Caines, et al. (2005). "Implementation of supply chain management and its impact on the value of firms." Supply Chain Management-an International Journal 10(3-4): 313-318.

Subramani, M. (2004). "How do suppliers benefit from information technology use in supply chain relationships?" MIS Quarterly 28(1): 45-73.

Van Groningen, L (2006). „Performance measurement in barge planning", Master Thesis, RSM Erasmus University

Wooldridge, M. and Jennings, N. R. (1995). "Intelligent agents: theory and practice." Knowledge management review 1995 (January): 1-62.

Wooldridge, M. (2005). Multi Agent Systems, John Wiley & Sons Inc., Chichester, England, 4th edition, 2005.