A distributed barge planning game

Martijn Mes, Maria-Eugenia Iacob, Jos van Hillegersberg

Centre for Telematics and Information Technology

University of Twente, Enschede, The Netherlands

Abstract. In this paper, we address a complex issue in the coordination of supply chains: the optimization of the alignment of barge and terminal operations in the Port of Rotterdam. Our research goal is to design a distributed multi-agent and service-oriented system architecture, which solves the barge handling problem through information exchange and a decision support system based on intelligent alignment of proposals. In order to validate this solution, we implement it by means of both a system and a management game, using web-service technology. We report our experiences with the game throughout its validation with domain experts.

1. Introduction

In this paper we address a complex issue in the coordination of supply chains, the optimization of the alignment of barge and terminal operations in the Port of Rotterdam, also called the barge handling problem. The Port of Rotterdam considers this problem as the most urgent problem in container barge hinterland transport. The poor alignment of barge and terminal operations leads to high direct costs and has even more indirect effects, such as environmental pressure, and congestion on the road and rail infrastructure. Solving the problem improves the hinterland connectivity and thereby the attractiveness of the port of Rotterdam significantly.

In the harbour, the 30 container terminals are visited daily by about 60 barges. Every barge visits on average eight container terminals to load and unload containers. For the handling of ships, appointments are made between terminal and barge operators. In the present situation, appointments are made by telephone, fax, and e-mail. Unfortunately, it happens frequently that appointments are not (or cannot) be met by either the barge or the terminal operator. In addition, the fact that barges usually visit multiple terminals creates dependencies between activities performed at different terminals. Thus, a disruption at one terminal can propagate through the port and disturb the operations of other barge and terminal operators. The result is that barge operators face uncertain waiting and handling times at terminals, and that terminals deal with uncertain arrival times of barges and idle time of their quay resources.

Besides the unreliability of appointments, the current 'manual' way of planning also results in strategic behaviour. For example, barges may announce a wrong number

of containers to handle in order to obtain more convenient time slots, and terminal operators respond by creating queues of barges to prevent idle times at their quays. This type of behaviour makes the alignment process even more uncertain and deteriorates the relationships between the terminal and the barge operators.

All the considerations above suggest that a distributed information system for the automated handling of the barge planning problem may significantly improve the situation, since such a system can increase the reliability of appointments, prevent opportunistic behaviour, and become and acceptable solution for all parties involved (as it will be explained in Section 2). Subsequently, the performance of both barges and terminals may improve. This is also the main contribution of this paper. Our research goal is to design a distributed multi-agent and service-oriented system architecture, which solves the barge handling problem through information exchange, and provides decision support based on intelligent alignment of proposals. We build on earlier research [1] in which a conceptual foundation (in terms of optimization algorithms) for this system was developed, evaluated, and demonstrated [2]. In order to validate and test the proposed architecture, we implement this architecture by means of both a system and a management game, using model-driven web-service technology. We also report our experiences during this game's validation with domain experts.

A series of very successful studies (e.g., [3], [4], [6]) have contributed to the recognition of the design science research methodology as valid IS paradigm. This methodology [6] distinguishes between five research activities, to which we adhere in this paper. With this section covering research activity 1 and 2, the remainder of this paper is organised as follows. In Section 2, we discuss the architecture of the future dynamic barge planning system (research activity 3). Section 3 presents the main contribution of the paper, i.e., the design and implementation of a management game based on the distributed system architecture presented in Section 2 (research activity 4). Section 4 (research activity 5) is concerned with the validation of this game (research activity 4). The paper ends with conclusions, and pointers to future work (Section 5).

2. The architecture

Before presenting the proposed architecture, we first summarize the most important requirements the future system must meet. It has to support the involved parties in optimizing their operations, and the conditions for adopting the new system have to be acceptable. Typically, for this problem we have to deal with different competing companies having different and conflicting objectives. In this business environment there is neither a clear hierarchy between companies nor dominant players. Also, no contractual relationships between terminals and barges exist, which means that no performance can be contractually enforced. In addition, this is a highly dynamic environment, i.e., information becomes known over time, and disturbances (may) take place. Due to the above characteristics, central coordination of all activi-

ties in the port (via a trusted third party or a control centre) is not an acceptable option for two main reasons. First, each party is reluctant to share information that could undermine its competitive position. Second, each party wants to have control on its own operations. Therefore, we have chosen for a distributed system in which the control of the process is assigned to the individual actors involved in the network. The proposed architecture (shown in Fig. 1) allows actors to plan their operations efficiently in such a way that the system is not unacceptable up-front: participants share only limited information and they keep autonomy over their own operations. The basic concept is a Multi-Agent System in which every party is represented by an agent (a piece of software). Together, the agents coordinate the barge and terminal activities in the port.

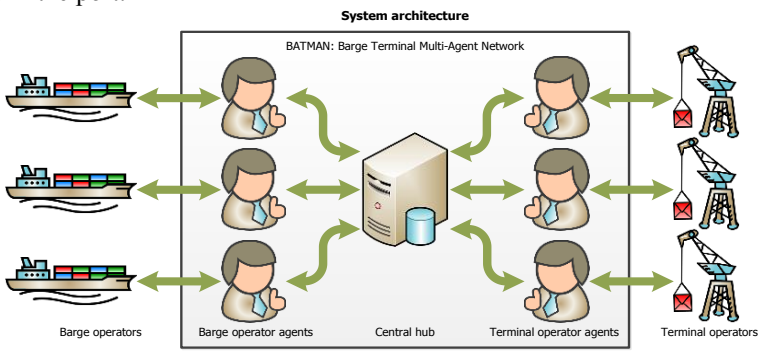

Fig. 1. Simplified system architecture

The sole core function of the hub is to facilitate the exchange and routing of messages between Barge Operator Agents (BOAs) and Terminal Operator Agents (TOAs). This exchange is carried out in two phases and is initiated by a BOA that has to plan a so-called *rotation*, i.e., a sequence of terminals to visit. In the first phase (Fig. 2, top) the BOA sends information about the numbers of containers that must be loaded/unloaded at each terminal in the rotation, and requests from all terminals a so-called *service time profile (STP)*. The STP calculated and issued by a TOA for a particular barge B and a terminal T, denotes the maximum departure time of B (after being serviced) from T, for every possible arrival time of B at T during a certain time period.

The idea behind using STPs is that terminals are not giving away competitive sensitive information (e.g., a high service time might indicate many visits at this terminal, but could also mean that there is no crew scheduled at this time). But still, it provides enough information for barges to plan their rotations efficiently.

Based on STPs received from all TOAs, the BOA calculates the best five rotations. For more information on the algorithm to compute the STPs and the algorithm to determine the optimal rotations using these STPs, we refer to [2]. The barge operator planner may choose one of the proposed rotations. Consequently, the BOA requests appointments with the respective terminals which constitutes the second phase in the planning process (Fig. 2, bottom). In case one or more terminals are not able to con-

firm the requested appointments (due to the fact that, in the meantime, appointments might have been made for other barges, which could invalidate previously issued STPs), the whole rotation is cancelled, and the process starts all over again, until a rotation can be completely booked. Due to space limitations we do not go here into further details regarding the exact design of the planning and interaction algorithms.

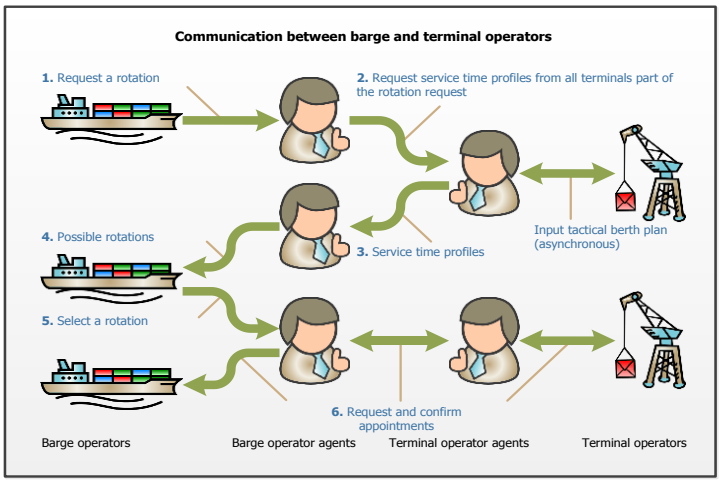

Fig. 2. The interaction protocol

The hub has been implemented by means of Enterprise Service Bus (ESB) middleware software, and has the following functionality: intelligent routing of messages, security (authentication and authorization of all agents), and monitoring. The agents (both TOAs and BOAs) have been implemented as web services.

3. The game

To gain acceptance (from both terminal and barge operators) for the designed system and to validate its functionality, we have developed a management game (available at www.bat-man.nl/game) in which we reuse the majority of the implemented system components. In terms of software architecture, the game version of the system has been slightly simplified in order to increase the interaction speed of the players within the system. Also, to make the gaming experience more pleasant and accessible, the game offers multiple types of user interfaces (laptop, tablet and smart phone).

The game is organized by a game master and played over the Internet. The game master has to configure the game and then invite a group of players to join the game. Within the game, the group can play multiple rounds where we can use different planning strategies. After playing all rounds, we can compare the scores of players within rounds, but also the overall scores (sum of scores within each round) for each

of the rounds. The latter comparison allows us compare the performance of different planning strategies. We now describe the steps that are to be carried out during a game in more detail.

1. Configure port. The port is divided into several regions called terminal groups. For each of these terminal groups, we set (i) the number of terminals each player has to visit within this group, (ii) the closing times, (iii) the handling times, and (iv) initial terminal utilization. Regarding the latter, it is not realistic to assume empty schedules for the terminals at the start of the game. Therefore, we randomly fill the terminal schedules with virtual appointments up to a given target utilization.

2. Invite players. We have a list suitable for up to 25 players, consisting of 25 barge names and icons. Either, we fill in the email address of all players after which they receive an invitation to join the game (with URL, username and password), or we simply number the players after which they login using this number.

3. Configure round. We have to set the following parameters: (i) duration in the game time (e.g., 3 days), (ii) actual duration of the game (e.g., 5 minutes), (iii) rounds to be played, (iv) planning mode, and (v) simulation mode. The last two settings will be explained later.

4. Summary. We observe a summary of all settings. If we agree to these settings we can start the game. Note that we can go back and forth between each of these steps as long as we didn't start the game. In addition we can save the game in each of these steps so we can replay the game some later time.

5. Planning phase. After logging in, players plan their rotation, which we explain later on, and confirm the plan of their choice. When all players confirmed their planning, the game master can end the planning phase thereby starting the simulation phase.

6. Simulation phase. In the simulation phase a clock is running and periodically all indicators, positions, performances, animation etc. are updated. On a map of the Port of Rotterdam (Fig. 3), we can follow the position of barges. In addition, the progress of all players is shown together with an estimate of their final score. Besides the map, there are other screens available with overview of terminal performance (utilization, fragmentation of appointments) and barge performance (handling time, sailing time, waiting time, all indicators are shown as planned, actual, and optimal).

7. Completion of a round. When all players completed their rotation or we are at the end of the simulation time, we finish the round and go to the scoring screen. Scoring of players is based on the difference in length of stay in the port between (i) their optimal plan (computed by the routing algorithm) and (ii) their actual rotation. We use this deviation from optimum as scoring criteria because the optimal length of stay in the port might differ per player. We keep track of this score in each round of the game. Within a round, we have a ranking of players based on this score. After playing all rounds, we display the score for all players over all rounds as well as the sum of scores for each player for each of the rounds. This way we can observe whether the availability of extra information results in higher scores. The game master can now start the next round and go back to the planning phase.

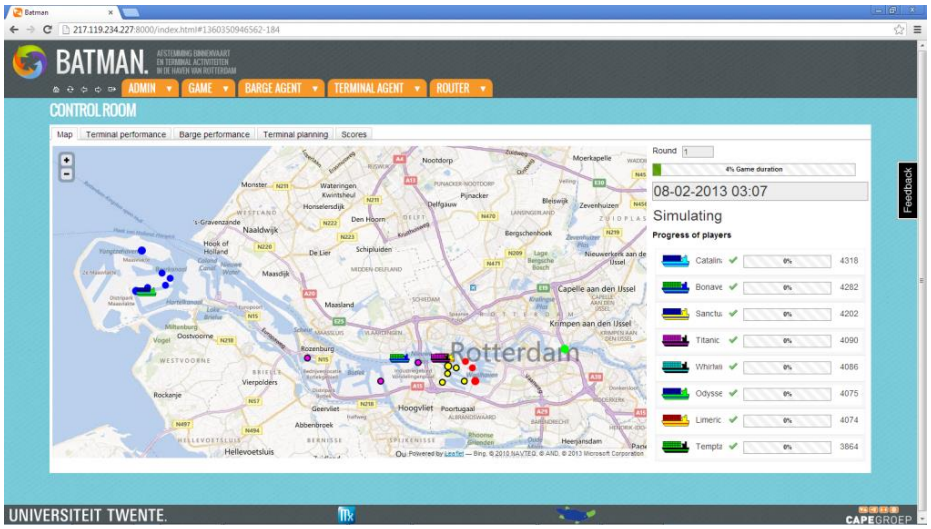

Fig. 3. Screen capture of the game

There are four types of rounds that can be played during a game and can be compared with respect to the planning performance. We describe them below.

1. The player choses the order in which terminals are visited without making an appointment with them, and without having information on their utilization.
2. The player choses the order in which terminals are visited and also makes appointments with them. As in type 1, no information on the utilization of these terminals is available. To make an appointment, the player asks the terminal whether it can handle the barge at a certain arrival time. The terminal's answer could be yes or no.
3. The player choses the order in which terminals are visited and makes appointments with these them. The player can now gain insight into the utilization of these terminals by requesting their STPs.
4. The player gets decision support from the barge operator agent by receiving a set of efficient rotations, together with various performance indicators of these plans. Now the player only has to decide which rotation to use.

By playing these four rounds, we demonstrate (i) more information on available terminal capacity results in better routes, (ii) the use of our decision support system results in the best routes, and (iii) overall increase in performance is achieved without central coordination; players maintain their autonomy.

4. Verification and validation of the game

The game has two main purposes (i) to familiarize students and professionals with the challenges of routing and scheduling problems and (ii) to illustrate possible new ways of planning within the port of Rotterdam. For both objectives, extensive validation of the game is important. We use several steps to come up with a valid game implementation. Here we make a distinction between the validation of the system architecture, of the algorithms, and of the game itself. The development methodology we followed and the game itself also provided opportunities for verification and validation. We used the SCRUM agile software development approach. The four weekly sprint review meetings were used to verify whether the implementation meets our requirements and obvious mistakes where found and placed on the product backlog to be fixed in a later sprint. The game itself is also a useful tool to validate the system architecture and the algorithms, as the game is basically a simulation layer on top of the original system. By playing the game, we observe the added value of the implemented algorithms and the user friendliness of our decision support system.

The validation of the application architecture focused on whether all required messages are sent and received, whether the response times are reasonable, and whether the application reacts properly on 'ill defined' messages and other types of exceptions. For this, we used the game with varying scenario's (planning horizon, number of players, number of terminal visits, terminal utilization, etc.). After extensive testing we conclude that the application architecture requires some small modifications for the gaming environment to reduce the response times. The reason for this is that in the game all players plan their rotation exactly at the same time, which is not likely to occur in practice. There are two algorithms used in the game: the barge operator agent uses a routing algorithm and the terminal operator agent uses an algorithm to compute the STPs. We implemented both algorithms as separate web services such that we can separately test them outside the game. In addition, we compared the output of our implementation to the original implementations of [2], who in turn benchmarked the algorithms against centralized planning methodologies.

To validate the game itself, we again used (i) manual testing under normal circumstances, (ii) extreme condition tests and (iii) sensitivity tests. Regarding the manual test, we play rounds with simplified settings such that we can compute the KPI's, such as the final game score, by hand and compare it with the results from the game. Extreme condition tests are carried out by simulating the model after setting certain variables (e.g., amount of players, number of visits per region, time-windows) to an extremely high or extremely low value and examining the behaviour of key variables. Results of these tests reveal evidence of high structural validity. Due to space limitations, the results of these tests are not shown in this paper. After the extreme-condition tests, a series of sensitivity runs were made to determine whether the sensitivity of the base model was within acceptable limits. The sensitivity tests were performed by playing with the same key variables, but varying them around given default settings (resembling reality). The results indicate that the model has a

meaningful and reasonable level of sensitivity to these parameters. Finally, we played numerous rounds with the project team, which also consists of several field experts, such as a terminal operator. In addition, these meetings were also visited by a representative of the port authority and a representative of a group of barge operators. We also organized gaming sessions at various national events, again with the presence of many field experts. After each round, we improved the game using feedback from the players. We are now ready to take the final and most important step, namely to organize gaming sessions for the target audience, i.e., the barge and terminal operators, and to assess their reactions.

5. Conclusion

In this paper we proposed a solution for the barge handling problem in the port of Rotterdam, which has been implemented in a distributed multi-agent planning system and a management game. The system consists primarily of several intelligent barge and terminal software agents and a hub that is responsible for the routing of messages between the interacting agents. The system architecture and the game have been tested and validated in several ways. The results of the tests indicate that the overall performance of the proposed solution is satisfactory.

We foresee two directions for further work. First, we want to improve the game itself by making the four planning options also available within the simulation phase. Instead of passively observing possible deviations from the plans (e.g., due to unforeseen waiting times), we can now change our plans dynamically. Second, we want to demonstrate the usability of our approach by organizing various gaming sessions with the target audience and by analysing the change in mental models of the players.

References

1. Douma, A., Schutten, M., and Schuur, P., Waiting profiles: An efficient protocol for enabling distributed planning of container barge rotations along terminals in the port of Rotterdam, Transportation Research Part C, vol 17, no 2, p. 133-148, 2009.
2. Douma, A. Schuur, P., and Schutten, J., Aligning barge and terminal operations using service-time profiles, Flexible Services and Manufacturing Journal, vol 23, no 4, p. 385-421, 2011.
3. Gregor, S., and Jones, D., Anatomy of a Design Theory, JAIS, vol 8, no 5, p. 312-335, 2007.
4. Hevner, A.R., March, S.T., and Park, J., Design research in information systems research, MIS Quarterly, vol 28, no 1, p. 75-105, 2004.
5. Malandraki, C. and Dial, R., A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem, European Journal of Operational Research, vol 90, no 1, p. 45-55, 1996.
6. K. Peffers, T. Tuunanen, M.Rothenberger, and S. Chatterjee, A Design Science Research Methodology for Information Systems Research, Journal of Management Information Systems, vol 24, no 3, p. 45-77, 2008.