Evaluation of a Multi-Agent Approach for a Real Transportation System

Wilfried Lepuschitz, Benjamin Groessing, Munir Merdan and Georg Schitter
Automation and Control Institute
Vienna University of Technology
Vienna, Austria
Email: {lepuschitz, groessing, merdan, schitter
@acin.tuwien.ac.at}

***Abstract*—Transportation systems represent the backbone of manufacturing systems and significantly influence their efficiency. Component failures or disturbances generally lead to traffic jams decreasing thereby the throughput and system performance if not even completely interrupt the production process. To cope with such situations, recently a multi-agent system for the transportation domain with capabilities for reconfiguration and adaptive routing has been introduced.**

The main contribution of this paper is to present the implementation of this multi-agent system on a real pallet transport system. The results of the experiments show the feasibility of the approach on a real manufacturing system and its capability for enhancing the system's fault tolerance preventing thereby a significant loss of production performance.

I. Introduction

Transportation systems are regarded as the backbone of manufacturing systems and offer many possibilities for capacity optimization. They can significantly influence the efficiency of the overall system [1]. The occurrence of traffic jams, caused by component failures or disturbances, may lead to a deviation from the initial production schedule, to degradations of the system performance or, in the worst case scenario, may even interrupt the production process. Furthermore, this can significantly increase production costs and consequently lower the economical competiveness of the product on the market. Due to this reason there is an increasing demand for more flexibility and reconfigurability in routing processes, which however significantly complicates the control of those systems. Nowadays the typical approach used in factories is a centralized global routing control with standardized path planning algorithms calculating all possible routing paths in advance. This is required as the programming languages for programmable logic controllers (PLCs) are generally not designed for carrying out such complex computations as path planning during operation [2].

The technology of multi-agent systems has been recognized as a powerful tool for developing highly flexible, robust, and reconfigurable industrial control solutions [3]. However, although confirmed as a promising approach and deployed in a number of different applications throughout the last few years, the widespread adoption of agent-based concepts by industry is still missing. Lack of awareness about the potentials of agent technology [4] and paradigm misunderstanding [5] due to the lack of real industrial applications, missing trust in the idea of delegating tasks to autonomous agents [6] as well as a "pioneer" risk, which accompanies every new technology that has not been proven in large scale industrial applications [7], are the main hindering factors for preventing the widespread adoption of agent technology in the manufacturing domain.

For showing the feasibility and benefits of multi-agent systems on real systems, this paper presents the outcome of long-run tests of a multi-agent system implemented on a transportation system. In order to measure the system's agility—the capability to react in a short period of time on the occurrence of unexpected disturbances—as well as introduced reconfiguration advantages the loss of productivity in the presence of disturbances is analyzed. The target transportation system at which the developed multi-agent system is implemented and tested in reality is the "Testbed for Distributed Control".

This paper is structured as follows. Section II discusses related works. In the third section we present in detail the control architecture introducing the target system, the automation agents and their capability for system reconfiguration. Section IV describes the test cases and Section V discusses the results. Finally the paper is concluded in the sixth section.

II. Related Works

Recent works address the problem of building confidence in agent-based systems with particulars techniques based on testing and monitoring or on simulation. An autonomic management execution system (MES) based on agents was desribed by Rolón and Martínez [8]. The MES was operated in a simulation environment for the production of numerous products using several reactor tanks and packaging units. The simulation results showed that the agent-based MES was able to keep the processing times of frequent orders within acceptable boundaries despite the failure of a broken packaging unit by rescheduling the orders to the remaining units using different strategies.

Bussmann and Schild reported the development of a flexible transportation system with an associated agent-based control within the frame of the Production 2000+ project [9]. The overall goal of the system is to continuously optimize the throughput while machine agents manage buffer sizes,

workpiece agents manage the processing state of a specific workpiece and shifting table agents try to optimize the routing. The system has been evaluated in a series of simulations based on real manufacturing data (product types, processing times, disturbance characteristics, etc.). The simulations have shown that the agent-based control is extremely robust against disturbances of machines as well as failures of control units achieving a performance near to the theoretical optimum. Moreover, the control system has been installed in the Daimler plant in Stuttgart validating the results of the simulations under real manufacturing conditions.

Another successful deployment of a multi-agent system by Rockwell Automation was the distributed control of a chill water system applied to reduce the manning as well as to improve readiness and survivability of US Navy shipboard systems [10]. A simulation environment as well as physical, table-top demonstrator was built in the Rockwell Automation research laboratories in Cleveland for testing the system's functionality in a set of scenarios that mimic the transactions of the real shipboard systems. The successful deployment of the Manufacturing Agent Simulation Tool (MAST) on a small scale transport system was reported by Vrba et al. [11] This conveyor system is used to transport palettes carrying raw materials or products between docking stations, where the palettes are held until a particular manufacturing process finishes. The palettes are routed by the diverters that use reachability knowledge to send the palette to the required docking station at the lowest cost, i.e. using the shortest possible path.

Besides the fact that the multi-agent systems presented above were implemented and their routing abilities tested in a real system presenting the advantages of the approach, a missing point is that these systems did not tackle any reconfiguration issue. The capability of the system to perform a reconfiguration of the control software in conjunction with a physical reconfiguration of the system in order to achieve functionality and efficiency is of vital importance. In previous work an agent-based approach combining local reconfiguration of control software with ontology-driven reasoning for improving performance and fault-tolerance was presented [12]. Ontologies are used to ensure the shared understanding among agents when exchanging information about their activities, which in this case directly reflects the configuration of their control software. Two simulation experiments were performed to analyze the impact of this approach on a transportation system's performance. In the first one only one agent, which is controlling an intersection, performs local diagnostics and reconfiguration. The second experiment analyzes the impact of local reconfiguration on the functionality of the global system. The results of the experiments in the simulation underline the increased reconfigurability, agility, robustness and fault tolerance of the transportation system based on agent technology. To show the feasibility and benefits of this approach on a real system, the experiments presented and evaluated in this paper represent the logical next step after simulation.

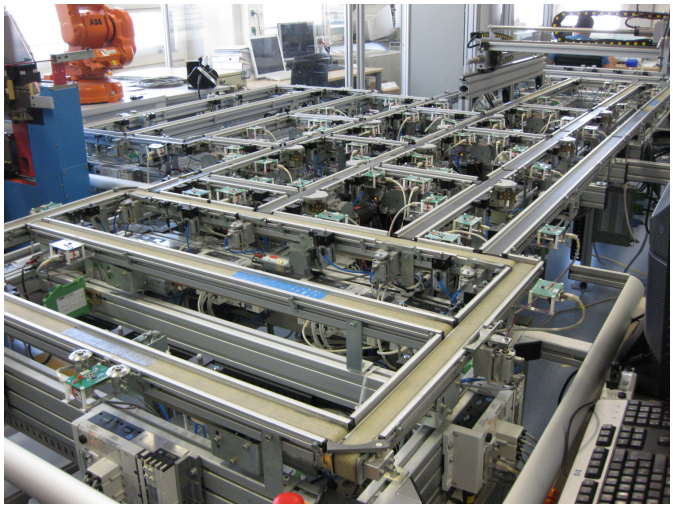

Fig. 1. Picture of the pallet transport system.

III. Control Architecture

The following sections introduce the target system and the applied control strategy using automation agents and their capability for reconfiguration.

A. Testbed for Distributed Control

The "Testbed for Distributed Control", located at the Odo Struger Laboratory of the Automation and Control Institute (ACIN), Vienna University of Technology, acts as the target system for the implementation and the performed tests. The testbed comprises workstations such as an industry robot, an automatic storage system with a handling unit for extracting parts and a portal robot for assembly tasks as well as a pallet transport system with redundant paths encompassing 45 conveyors with 32 intersections and a set of indexstations with grippers for holding pallets (see Figure 1).

Within this system, the major transport tasks are the delivery of a part from the storage to a machine, to carry an unfinished sub-assembly between the machines, and to remove the finished product from the system. A main objective is to transport pallets between index stations in a minimum amount of time, usually following the shortest path and avoiding broken components and congested ways. As the presented tests were focused on the routing of pallets, only the pallet transport system is described in the following in more details.

The main mechatronic components of the pallet transport system are:

- *Conveyor belts*, which move pallets from one place to another.
- *Indexstations*, which comprise a gripper for holding a pallet in a defined position, so that a workstation can process its content.
- *Intersections*, which represent a connection between several conveyors. Depending on the direction of its adjacent conveyors, an intersection operates either as a *diverter*, which receives pallets from one input conveyor and routes

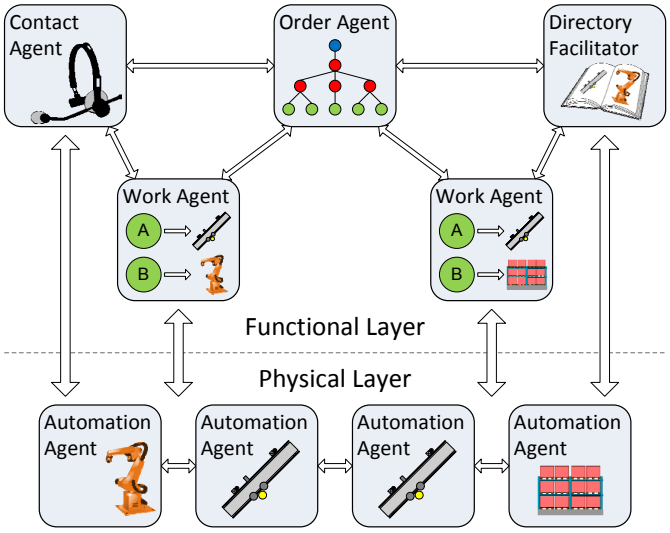

Fig. 2. Agent types of the multi-agent system.

them to one of its output conveyors according to their target destinations, or as a *junction*, which receives pallets from two input conveyors and feeds them to one output conveyor according to their priorities.

The indexstations and intersections also incorporate sensors for detecting pallets and blockers for stopping them. To identify the pallets and their destinations, the indexstations and intersections rely on Radio Frequency Identification (RFID) tags, which are accessed through RFID readers. Furthermore, grippers, switches and blockers are equipped with sensors for verifying their momentary state.

A set of 38 embedded controllers of the type CPX-CEC-C1 by Festo is employed to control the indexstations and intersections as well as the conveyors. Each controller comprises an Xscale-PXA255 agile Intel microprocessor with 400MHz, 28 MB Flash and 24 MB RAM and several I/O-modules (digital inputs, digital outputs and valves) for connecting the sensors and actuators. On the contrary, every RFID module is controlled by a small embedded controller of the type Digi Connect ME.

B. Automation Agents and the Multi-Agent System

The multi-agent system controlling the testbed is based on the Jade [13] platform for agent management and inter-agent communication. The system is divided into two layers: the functional layer and the physical layer (see Figure 2). The function layer contains a set of different functional agents, which carry out higher level functions such as system management (contact agent), the scheduling of jobs (order agent) or the distribution of tasks (work agent). To avoid bottlenecks, the functional agents may also exist in redundant forms. Detailed information about the provided functionality of these higher level agents can be found in [4], [14].

Each component such as an intersection is represented and controlled by an automation agent and thereby embedded into the multi-agent system in its physical layer. Accordingly

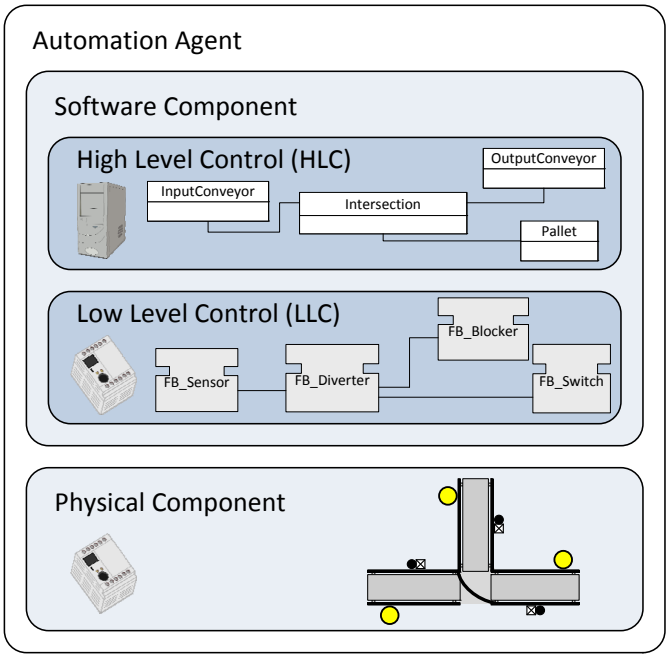

Fig. 3. Architecture of an Automation Agent.

the physical component can be regarded as each agent's embodiment and the software acts as the agent's intelligence. The automation agents register their offered services to a functional agent of the type directory facilitator.

To facilitate the design of a multi-agent control system in the manufacturing domain, a generic agent architecture for the automation agents was introduced [15]. As can be seen in Figure 3, this architecture clearly separates the control software into the high level control (HLC) and the low level control (LLC) as it is widely agreed to split agent-based manufacturing control into these two layers [16]. For the communication between the two layers an interface was developed to easily exchange data [17].

The HLC, implemented in Java™, controls the behavior of the automation agent in conjunction with the other agents of the system. It comprises a world model repository for storing a symbolic representation of the surrounding environment. The world model is used by the agent to reason about the states of the world and to perform diagnostic tasks by detecting inconsistencies with the information received from the environment (i.e. sensors and other agents). In general the world model is updated after recognizing changed world conditions or after performing actions. The well-disposed reader may find more details about the world model in [15].

The hardware-near LLC provides the basic functionality of a component, i.e. routing a pallet in the case of an intersection operating as a diverter, and has access to the sensors and actuators. It comprises a limited set of reactive behaviors, collects and processes the information from sensors and based on the result performs particular actions or informs the HLC about significant events. The LLC was developed using the 4DIAC-IDE [18] for designing control applications based

on the industrial standard IEC 61499 [19]. Both the Festo CPX and Digi Connect ME controllers host instances of the FORTE, which is a small portable C++ runtime environment for running IEC 61499 applications on embedded control devices.

C. System Reconfiguration

Based on Dijkstra's algorithm [20], a shortest path algorithm was implemented, which is used for calculating routing tables for each intersection [21]. Furthermore, an algorithm was developed which is used by higher level agents of the type contact agent (CA) for estimating the reachability of all destinations (i.e. in general the workstations for manipulating the pallets' loads) in the system which is important in the case of component breakdowns. In the case of unreachable destinations, this so-called change direction algorithm (CDA) determines necessary direction changes of specific conveyors [22].

In the case of failures, such as a breakdown of a component or a stuck pallet inside an intersection, the system reacts as follows:

1) The failure is detected either by the automation agent's LLC using the sensors or by the HLC using its world model for determining inconsistencies [23].
2) Based on the given information the automation agent informs a CA which starts the CDA and compares its results with the actual system state.
3) In the case that the CDA recommends a new configuration, the CA updates its ontology, requests the automation agents of the concerned conveyors to change their directions and informs the adjacent intersection automation agents to update their world model.
4) After updating their world model, the intersection automation agents start reconfiguration mechanisms for adjusting the components' functionality according to the new conditions (e.g. a junction now becomes a diverter) [24].
5) According to the changed conveyor directions, the routing tables of the intersection automation agents are updated to route pallets to their appropriate output conveyors.

IV. Test cases

In order to investigate the effectiveness of the system reconfiguration using the CDA and local reconfiguration of control software in the case of a detected failure, a set of experiments with the pallet transport system is conducted. The process flow for the performed tests consists of two process steps. At first the pallets start at a workstation at indexstation D1 where they have to be processed for a period of 10 seconds. As a second step they need to be processed at another workstation at a different indexstation D2 for a longer period than at D1, which is 20 seconds. In order to avoid a bottleneck at this second workstation, a redundant workstation is used at indexstation D3 within the testbed. Hence, each of the redundant two workstations only needs to process half of the pallets in the system. After the second step, each pallet shall return to the workstation at D1 to start another process cycle.

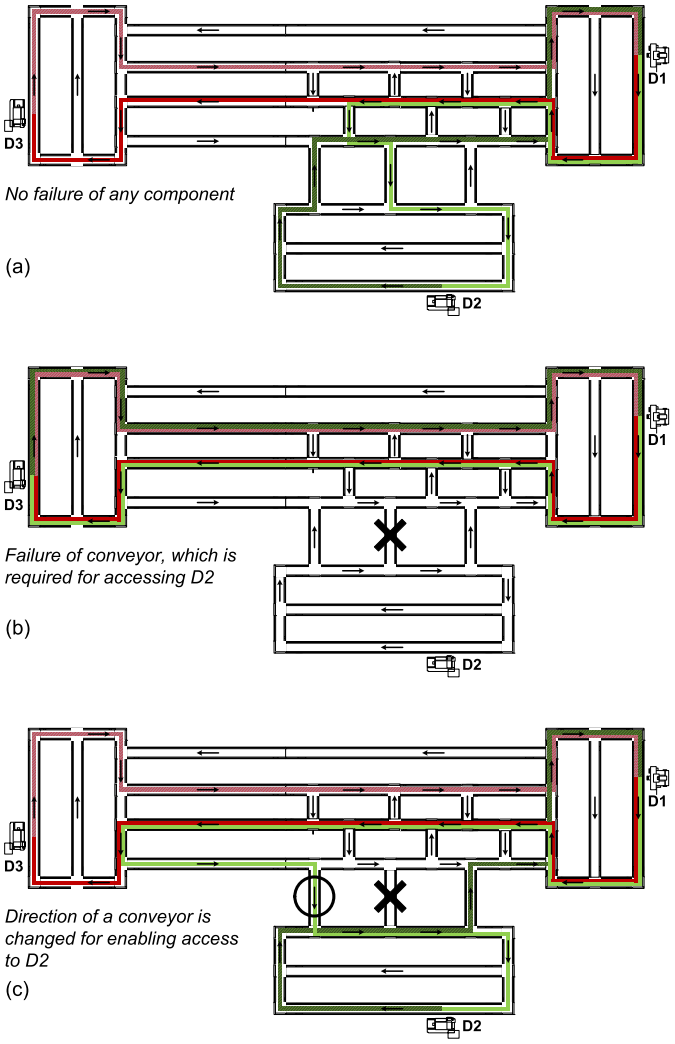

Fig. 4. Layout of the pallet transfer system.

Three test cases are performed with a number of 10 pallets in the system:

(a) Test case without conveyor failures: All destinations are reachable. Figure 4a shows the shortest paths the pallets should take. At D1, half of the pallets receives D2 as next destination while the other half is sent to D3. Therefore, those pallets with D2 as next destination should follow the green path and afterwards return to D1 along the shaded green path. On the other hand, those pallets with D3 as next destination should follow along the red path to D3 to return afterwards along the shaded red path to D1.
(b) Test case, when one critical conveyor fails: Indexstation D2 is not reachable. Hence, all pallets need to be sent to D3 after being processed at D1. It can be seen in Figure 4b that both the green and red path are similar as all pallets have to move from D1 to D3 and back.
(c) Test case, when one critical conveyor fails but a resolving solution is found by the system: The system is recon-

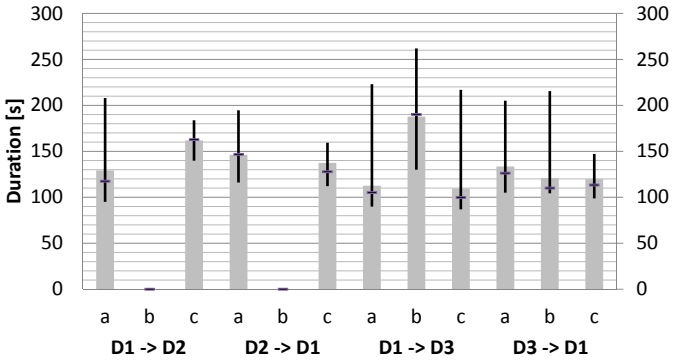

Fig. 5. Path durations for all test cases.

figured using the CDA and indexstation D2 is therefore accessible again. Figure 4c shows that the red path for one half of the pallets to indexstation D3 and back is unchanged, but the green path to indexstation D2 and back to D1 for the other half of the pallets is different and a bit longer compared to the one in test case (a).

The following assumption is defined in the context of the perfomance tests: Applying the CDA and system reconfiguration mechanisms enhances fault tolerance in the case of a component failure leading to a throughput comparable to the case without failures.

V. Discussion of Results

In order to gain measurement data, all indexstations log the processed pallets using their ID and a time stamp. This allows the calculation of the following characteristic numbers:

- Throughput per hour, which represents the number of produced products per hour in the manufacturing system;
- Duration of a complete process cycle starting at indexstation D1 until a pallet returns to D1 for beginning the next process cycle;
- Travel duration on a path from one indexstation to a particular other indexstation, which includes also waiting times at intersections and indexstations due to other pallets.

Figure 5 depicts the travel time of the pallets between the indexstations representing therefore the duration of the paths for all test cases. Each grey bar represents the average duration of a certain path (e.g. the path from D1 to D2) for a given test case. The vertical black stroke shows the spread from the minimum to the maximum value and the small horizontal black stroke represents the median value of a path's duration.

It can be seen that in test case (a) the returning path from D2 and D3 to D1 takes in average a longer time of about 20 seconds. This can be explained with occuring traffic jams at D1 when pallets return from D2 and D3 at the same time.

As indexstation D2 is not reachable in test case (b), there is no data concerning paths to and from D2. The duration for the path from D1 to D3 is obviously significantly longer than in the other test cases. This is evident, as D3 represents a bottleneck in this test case due to the longer processing time

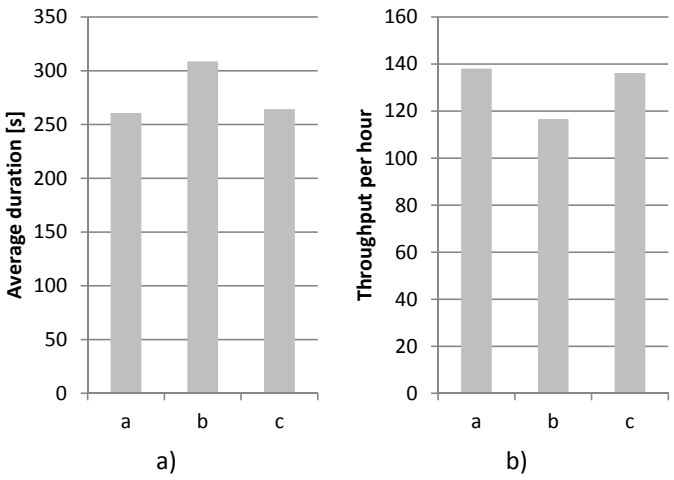

Fig. 6. Average duration of a process cycle (a) and throughput per hour (b) for all test cases.

leading therefore to serious traffic jams. On the other hand, the return path to D1 takes in average a slightly shorter time than in test case (a) as the pallets return from D3 with enough time difference leading therefore to no traffic jams at D1.

As the path in test case (c) from D1 to D2 is longer, so is the travel time, which can be clearly seen in the diagram. The path from D1 to D3 takes roughly the same time as in test case (a). However, the returning paths from D2 and D3 to D1 are slightly shorter, which might be explained by less traffic jams at D1 due to a longer path to D2 resulting in a better distribution of the pallets on the testbed.

Figure 6a shows the average duration of one process cycle for 10 pallets which encompasses travel times and processing times. While in test case (a) the processing time is in average 260.7 seconds, test case (b) reveals a processing time of 308.7 seconds resulting therefore in an increase of 18.4%. Due to this, the average throughput per hour with 10 pallets drops from 138.1 pieces to 116.6 pieces as can be seen in Figure 6b, which is a loss of 15.6%.

On the other hand when the system is reconfigured by the agents to restore the reachability of D2, one process cycle takes in average 264.4 seconds, which is an increase of only 1.4% compared to the failure free test case (a). With 136 pieces per hour the throughput is correspondingly a bit lower compared to test case (a) by only 1.4%. Thus, the application of our approach significantly improves the system efficiency compared to a conventional system without reconfiguration capabilities. Therefore, the assumption stated at the end of Section IV is confirmed by the measurements performed on the real system.

However, evidently the results depend on certain system parameters. As mentioned in Section IV, the processing time at D1 is 10 seconds while at D2 and D3 it is 20 seconds. Tests with shorter processing times result in only rare traffic jams for the test cases (a) and (c) when pallets return to D1 at the same time and no traffic jams in test case (b). Hence, in this case a better routing strategy would be to send

all pallets to only one of the redundant workstations for the second process step. For 10 pallets, the size of the testbed is large enough so that the pallets are evenly distributed on the paths between two indexstations. However, in the case of more pallets in the system, traffic jams would certainly occur at single workstations despite the small processing time.

On the other hand a longer processing time than the one in the given test cases will likely lead to more traffic jams. Hence, in such a case the availability and reachability of a redundant workstation is of utmost importance to keep up the system's throughput.

VI. Conclusion

This paper presents the implementation and evaluation of a multi-agent system on a real pallet transport system. The automation agents are able to observe their environment by using information from sensors or from other agents of the system. Hence, failures can be detected by the agents and due to the conjunction of local reconfiguration of control software with agent technology, the system is able to react adequately on these disturbances.

The results of the experiments clearly show the benefits of the multi-agent system in the case of a conveyor breakdown. Identifying an alternative route and reconfiguring the system accordingly results in a better performance for most cases. However, if the processing time is too short and the number of pallets in the system is rather small, jams will not occur even though certain destinations of the system are unreachable making therefore such a reaction obsolete. On the contrary, in the case of an unreachable system part and longer processing times or a higher number of pallets, a direction change of a conveyor significantly increases the performance back to the failure free case due to the regained reachability of those destinations.

Further research work will be concerned with using likewise mechanisms for avoiding traffic jams even in a failure free case by routing pallets on alternative routes or by changing conveyor directions to enable such routes. Also, the distribution of pallets to redundant workstations could be optimized to increase the throughput of the system. It is also planned to adapt the previously used simulation environment to the actual layout of the testbed to carry out further long-run simulation experiments and compare their results with the outcome of the work presented in this paper.

Acknowledgment

The authors acknowledge the financial support from the BRIDGE program by the Austrian Research Promotion Agency under contract FFG 829576 as well as from the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreement No. 284573.

References

[1] M. D. Byrne and P. Chutima, "Real-time operational control of an fms with full routing flexibility," *International Journal of Production Economics*, vol. 51, no. 1-2, pp. 109–113, 1997.

[2] P. Vrba and V. Mařík, "Capabilities of Dynamic Reconfiguration of Multiagent-Based Industrial Control Systems," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 40, no. 2, pp. 213–223, 2010.

[3] P. Leitao, "Agent-based distributed manufacturing control: A state-of-the-art survey," *Engineering Applications of Artificial Intelligence*, vol. 22, no. 7, pp. 979–991, Oct. 2009.

[4] M. Merdan, "Knowledge-based Multi-Agent Architecture Applied in the Assembly Domain," PhD Thesis, Vienna University of Technology, 2009.

[5] G. Candido and J. Barata, "A Multiagent Control System for Shop Floor Assembly," in *3rd International Conference on Industrial Application of Holonic and Multi-Agent Systems*, 2007, pp. 293–302.

[6] K. P. Sycara, "Multiagent Systems," *AI Magazine*, vol. 19, no. 2, pp. 79–92, 1998.

[7] M. Pěchouček and V. Mařík, "Industrial deployment of multi-agent technologies: review and selected case studies," *Autonomous Agents and Multi-Agent Systems*, vol. 17, no. 3, pp. 397–431, 2008.

[8] M. Rolón and E. Martínez, "Agent-based modeling and simulation of an autonomic manufacturing execution system," *Computers in Industry*, vol. 63, no. 1, pp. 53–78, 2012.

[9] S. Bussmann and K. Schild, "An agent-based approach to the control of flexible production systems," in *Proc. of the 8th IEEE Int. Conf. on Emergent Technologies and Factory Automation (ETFA 2001)*, 2001, pp. 488, 481.

[10] F. P. Maturana, P. Tichý, P. Šlechta, F. Discenzo, R. J. Staron, and K. Hall, "Distributed multi-agent architecture for automation systems," *Expert Systems with Applications*, vol. 26, no. 1, pp. 49–56, 2004.

[11] P. Vrba, V. Mařík, and M. Merdan, "Physical deployment of agent-based industrial control solutions: MAST story," in *Proceedings of the IEEE International Conference on Distributed Human-Machine Systems*, Athens, Greece, 2008.

[12] M. Vallée, M. Merdan, W. Lepuschitz, and G. Koppensteiner, "Decentralized Reconfiguration of a Flexible Transportation System," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 3, pp. 505–516, 2011.

[13] "Jade - Java Agent Development Framework," access date September 2012. [Online]. Available: http://jade.tilab.com

[14] M. Merdan, T. Moser, P. Vrba, and S. Biffl, "Investigating the robustness of re-scheduling policies with multi-agent system simulation," *International Journal of Advanced Manufacturing Technology*, vol. 55, no. 1–4, pp. 355–367, 2011.

[15] M. Vallée, H. Kaindl, M. Merdan, W. Lepuschitz, E. Arnautovic, and P. Vrba, "An Automation Agent Architecture with A Reflective World Model in Manufacturing Systems," in *IEEE International Conference on Systems, Man, and Cybernetics*, San Antonio, Texas, USA, 2009, pp. 305–310.

[16] J. Christensen, "HMS/FB architecture and its implementation," in *Agent-based manufacturing: Advances in the Holonic Approach*, S. M. Deen, Ed. Springer-Verlag, Berlin, Germany, 2003, pp. 53–88.

[17] W. Lepuschitz, M. Vallée, M. Merdan, P. Vrba, and J. Resch, "Integration of a Heterogeneous Low Level Control in a Multi-Agent System for the Manufacturing Domain," in *IEEE International Conference on Emerging Technologies and Factory Automation*, 2009, pp. 1–8.

[18] 4DIAC-Consortium, "4DIAC - Framework for Distributed Industrial Automation and Control, Open Source Initiative," access date September 2012. [Online]. Available: http://www.fordiac.org

[19] IEC 61499-1, *Function blocks – Part 1: Architecture*. Geneva: International Electrotechnical Commission, 2005.

[20] E. W. Dijkstra, "A note on two problems in connexion with graphs." *Numerische Mathematik*, vol. 1, p. 269271, 1959.

[21] M. Merdan, G. Koppensteiner, I. Hegny, and B. Favre-Bulle, "Application of an ontology in a transport domain," in *IEEE International Conference on Industrial Technology*, 2008, pp. 1–6.

[22] G. Koppensteiner, M. Merdan, I. Hegny, and G. Weidenhausen, "A change-direction-algorithm for distributed multi-agent transport systems," in *IEEE International Conference on Mechatronics and Automation*, 2008, pp. 1030–1034.

[23] M. Merdan, M. Vallée, W. Lepuschitz, and A. Zoitl, "Monitoring and diagnostics of industrial systems using automation agents," *International Journal of Production Research*, vol. 49, no. 5, pp. 1497–1509, 2011.

[24] W. Lepuschitz, A. Zoitl, M. Vallée, and M. Merdan, "Towards self-reconfiguration of manufacturing systems using automation agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 41, no. 1, pp. 52–69, 2011.