Phase Agents and Dynamic Routing for Batch Process Automation

Wilfried Lepuschitz[1], Benjamin Groessing[1], Emilian Axinia[2], and Munir Merdan[1]

[1] Automation and Control Institute, Vienna University of Technology, Vienna, Austria
{Lepuschitz,Groessing,Merdan}@acin.tuwien.ac.at
[2] COPA-DATA GmbH, Salzburg, Austria
EmilianA@copadata.com

Abstract. Currently applied process automation solutions rely on predefined control recipes with preprogrammed material transfer routes in the subjacent control software. Thus, the flexibility is limited with regard to dynamic environment conditions such as a change of production job priorities or a modification of the system layout. In this context, agent technology is seen as a promising approach for providing such flexibility. This paper presents a multi-agent system for batch process automation, which introduces the concept of phase agents for controlling the physical equipment. The phase agents incorporate control software based on the standard IEC 61131 for PLC programming in consideration of compliance to the standard IEC 61512 Batch Control. In the context of material transfers, a route finding algorithm is introduced for dynamically calculating suitable routes. Moreover, demonstration applications are presented to show the feasibility of the approach.

Keywords: Agent technology, batch process, flexible automation, dynamic routing.

1 Introduction

Current process automation solutions possess a limited capability concerning agile adaptation to internal and external disturbances. The applied traditional scheduling approaches with their rigid and centralized control structures suffer from the lack of flexibility and configuration abilities especially if unexpected events occur, such as the unavailability of resources or sudden changes in task priorities [1,2]. The lack of adaptability can therefore result in deviations from the initial working plans causing significant time and financial losses [3]. Considering the fact that resources have limited capacity over an observed time period, the application of techniques for workload balancing is essential to avoid workflow bottlenecks. The selection of a resource with the lowest workload for the assignment of a new job from the list of unfinished jobs can maximize the system throughput, while minimizing work in process and lowering the level of operating expense [4].

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 37–48, 2013.

Furthermore, material transfers between destinations in a complex pipe system have to be organized considering the current system state as the needed transportation time could significantly influence the efficiency of the overall system. Regarding the complexity of these systems and their dynamic nature (e.g. a component breakdown can cause that parts of the systems are not available), the process of choosing the best route at a specific moment can be difficult. If a production system incorporates a high routing flexibility, then during the breakdown, repair, or maintenance of a required resource, materials can be rerouted dynamically to other appropriate resources for processing the particular product [5]. However, the typical approach used in today's factories is based on a centralized global routing control with standardized path-planning algorithms for constituting the routing paths in advance [6]. A piping and instrumentation diagram (P&ID) of an existing winery's storage tank system, which consists of 60 tanks, a set of pumps and numerous valves, is presented in [7]. The currently used automation solution for the given use-case involves 1770 manually hard-coded routes for each combination of source and destination tank. Evidently, a modification of the tank system, e.g. the integration of an additional tank, requires modifying the implemented set of routes in the programmable logic controller (PLC). In a matrix of 60 tanks with 1770 routes, installing a new tank requires 60 additional routes to be implemented and therefore 1830 routes in total. Applying changes to the pipe layout between the tanks results in even more complex and time consuming efforts as the already implemented programs realizing the routes have to be reprogrammed accordingly.

To overcome the limitations of current automation solutions, the introduction of artificial intelligence techniques is seen as a promising trend in the process industry [8]. In this context, multi-agent technology is recognized as a powerful tool for developing highly flexible, robust, and reconfigurable industrial control solutions [6,9]. It offers a convenient way to cope with the dynamics in large complex systems, making the control of the system decentralized, thereby reducing the complexity, increasing flexibility, as well as enhancing fault tolerance [10]. Hence, agent technology is proposed for usage in the process domain according to an analysis of its advantages and disadvantages as presented in [11].

For improving the performance of batch process systems, this paper introduces a multi-agent system with dynamic scheduling and routing strategies. The schedule is not calculated in advance, but determined due to the dynamic negotiations between the agents. Considering the significance of route planning for the batch process domain, a dynamic route finding algorithm is presented for improving the flexibility of the system. Besides, the agent-based system is designed to be compliant to the commonly applied industrial standards IEC 61131 [12] for programmable logic controllers (PLC) and IEC 61512-1 Batch Control [13].

This paper is structured as follows. Section 2 briefly introduces the standard IEC 61512. The agent system architecture is presented in Section 3 and the dynamic route finding algorithm is detailed in Section 4. Section 5 is concerned

with two implementation examples of the given approach. Finally the paper is concluded in Section 6 with a summary and an outlook on further tasks.

2 IEC 61512 Batch Control

The standard IEC 61512-1 Batch Control—Part 1: Models and Terminology (IEC 61512) [13], respectively its counterpart ISA S88.01 Batch Control, introduces "a framework for the specification of requirements for batch process control, and for their subsequent translation into application software" [14]. The standard is widely accepted in the industry and currently applied batch management systems utilize it as the basis for their structural models to ensure a certain grade of comparability and interoperability [15].

IEC 61512 provides reference models and structures as well as definitions concerning processes (process model), physical equipment (physical model) and control software (procedural control model) in the domain of process automation. Generally, the introduced hierarchical models comprise four significant layers, such as procedure, unit procedure, operation and phase in the case of the procedural control model. In this context, actual process functionality is achieved by mapping elements of the procedural control model onto those of the physical model.

Concerning the equipment entities and procedural elements, a general concept for their operational modes and states is presented. The momentary operational mode of an entity (automatic, semi-automatic or manual mode) constitutes its execution behavior especially regarding the extrinsic influence on changing its operational state. Concerning the operational state, the standard provides a model of a state machine containing a set of states and according commands to trigger transitions between the states (see Fig. 1). If an entity is in a quiescent or final state, it performs currently no operation and waits for the next command. On the contrary, during a transient state the entity makes use of sensors and actuators to provide some kind of process functionality. In such a state a transition is triggered either by an extrinsic command as well or by the entity itself after fulfilling specific conditions (e.g. reaching a safe state of the process in the state *stopping* or completing a process task in the state *running*).

The general strategy for the execution of a process is described within the concept of recipes. Control recipes, which are stored and managed using a PC-like batch server, constitute in a hard-coded manner the physical equipment to be used and are linked with the according control software hosted on controllers such as PLCs. Theoretically, the standard allows this connection on any of the four described layers of the structural models, but commonly the industrially applied batch control systems apply the linkage of control recipe and control software on the phase layer of the procedural control model with corresponding equipment modules of the physical model [14]. A so-called phase logic interface is employed for performing the actual connection. The process steps of the control recipe, which determine the execution sequence of the production process, are converted into commands for the linked phases and their state machines in the

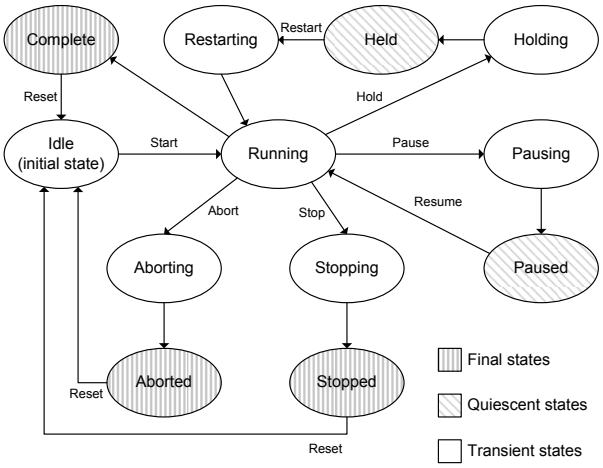

Fig. 1. State machine model of the standard IEC 61512 [13]

PLCs. Furthermore, status information is thereby sent from the PLCs about the current status of the phases back to the batch server.

3 Agent System Architecture

Process automation represents a complex domain whose functions are composed of a mixture of physical and non-physical components. In this context, a process automation system can be arranged in a "hierarchical" set of layers as it is commonly done in current industrial solutions. Figure 2a depicts the layers of a recently introduced industrial batch management system in the framework of the zenon software developed by COPA-DATA. To facilitate its acceptance in the industry, this system is generally compliant to the standard IEC 61512 (see Section 2).

Using the equipment editor, the user specifies the physical components and PLCs with the provided phases for their usage in the batch management system. The recipe editor of the zenon editor is employed for creating master recipes, which incorporate the basic process steps for producing specific products without referring to any distinct equipment or control units. The zenon runtime is then used for deriving the control recipe from the master recipe by linking the recipe phases with the corresponding equipment phases. Finally, the control recipe is executed with the recipe execution system of the zenon runtime, which actually produces a batch.

Based on the analysis of the industrial batch management system, four layers can be specified: management, planning, scheduling and executive layer (see Figure 2b). Accordingly, different types of agents are employed for realizing the functionality of these layers (see Figure 2c).

The management layer is responsible for keeping track of the entire functionality of the system and provides a communication interface with the external

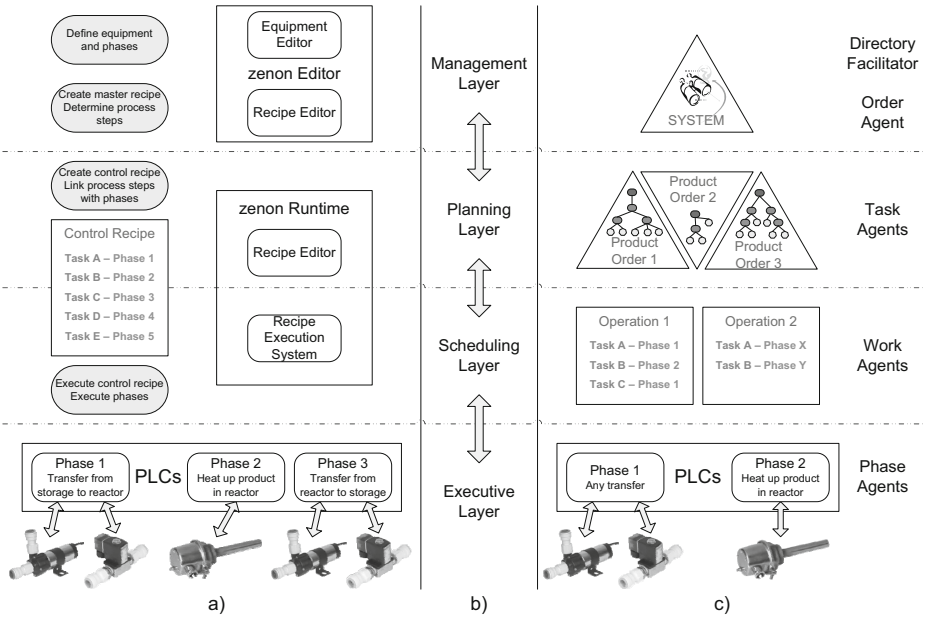

Fig. 2. Structure of an industrial batch management system (a), specified layers (b), and architecture of the MAS (c)

environment. The Directory Facilitator Agent (DFA) manages a list of the process functionalities, which are provided by the Phase Agents (PA) controlling the physical equipment. The Order Agent (OA) represents an interface to human operators, which can be used for generating product orders. In this case the OA looks up the according recipe and creates a job comprising product type, amount and job ID. Subsequently, the job is delegated to a Task Agent (TA), which is in charge of the recipe execution for producing the batch.

The planning layer is reponsible for determining the appropriate phases and thereby the equipment and resources, which can be employed for the recipe execution. The Task Agent chooses the phases for the recipe execution by searching for PAs providing suitable services in the list of the DFA. As soon as the appropriate phases are identified, the tasks for the first operation of the recipe are created and sequentially delegated to a Work Agent (WA). Such an operation involves the actual production tasks (such as heating up material in a reactor tank) as well as transfer tasks between tanks and reactors with a given amount of material to be transported. Upon completion of all tasks of an operation, the tasks for the following operation are created and delegated. After finishing all operations and thereby the production of the batch, the TA informs the OA.

The scheduling layer is responsible for negotiating with the resources and for the according task allocation. Work Agents govern the execution of production and transfer tasks. In the case of the latter, this involves the dynamic calculation

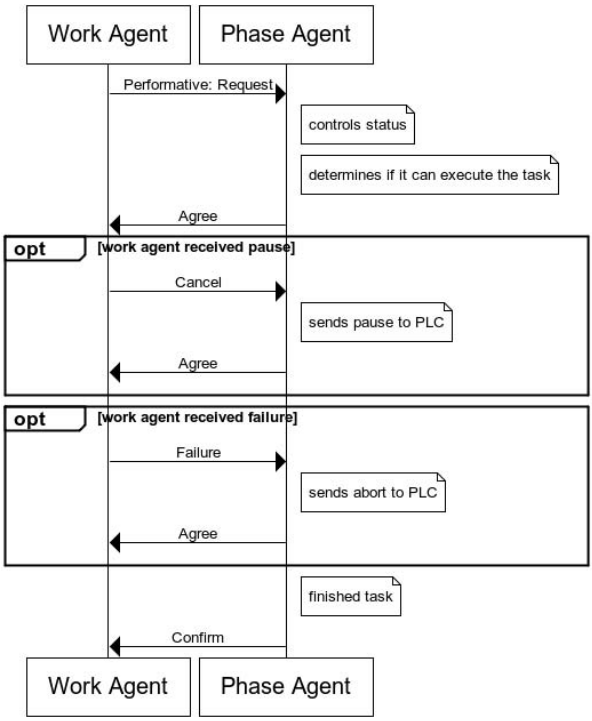

Fig. 3. Example communication between work agent and phase agent

of a route through the system by employing a route finding algorithm, which is described in Section 4. For issuing the execution of a task, commands are sent to the corresponding PA(s) to induce the activation of its (their) provided services (see Figure 3). In the case of a transfer task, the calculated route is also sent to the PA(s) in charge of the transfer equipment (i.e. pumps and valves). Upon completion of the task, the WA notifies the TA for receiving the next task.

The execution layer is in direct control of the production system's equipment. On this layer, the production tasks are executed and if a failure or disruption is diagnosed, the superjacent layers are informed. Each Phase Agent controls a set of physical components for providing the functionality of a phase, such as heating up material in a designated reactor tank to a certain temperature. Regarding the provided phases with their functionality hard-coded in the PLCs, each PA represents a wrapper around a distinct phase acting thereby as an interface between the according IEC 61131 program and the agent framework. Hence, also existing industrial solutions based on phases could be integrated into an agent-based production system. Compared to these phases realizing the actual production tasks, the PAs in charge of transfer equipment are able to open the valves and activate a distinct pump dynamically according to the route received from a WA in the case of a transfer task.

While current industrial solutions constitute the used equipment hard-coded in the control recipes, the presented agent system architecture allows the dynamical allocation of jobs to the process cells.

4 Route Finding Algorithm

For dynamically calculating the route from one component of the system to another one, a route finding algorithm based on Dijkstra's algorithm [16] is invoked. An ontology containing the revelant information about the system components and the way they are connected to each other is used in an Extensible Markup Language (XML) format as a representation of the physical equipment. Since Dijkstra's algorithm relies on nodes and directed weighted edges, it is necessary to map tanks, valves, pumps, pipes and other components to an according graph data structure (see Figure 4).

Pipe segments with a positive length number are directly used as edges. Welding points generally represent the pure crossing of pipe segments while connection points are commonly applied for the linkage of pipes with components such as pumps and valves. A direction information can also be applied to the connection points, which is essential for defining allowed flow directions through a pump or the exit and entry points of tanks. Furthermore, it is possible to define complex components denoted as topology, which themselves consist of sub-components (e.g. a crossing valve for three or more pipe segments modelled by using a set of basic valves). Path restrictions in the form of a white list (allowed paths) or black list (forbidden paths) may also be defined for these topologies. Finally a media-media compatibility matrix and a material-media compatibility matrix constitute if liquids (media) are compatible with each other concerning consecutive liquid transfers and if they might be transferred through pipes made of specific materials.

The components are linked to a type specific set of variables, which are required for incorporating the following route finding criteria:

- Avoid current transfers: Components that are currently in use by another active transfer may not be considered.
- Avoid reserved routes: Components that have been marked as reserved for a later transfer may not be considered.
- Use functional state: Components that are physically in a disallowed state may not be considered.
- Use service state: Components that are marked as blocked for maintenance may not be considered.
- Use medium-medium compatibility matrix: Uncleaned components, which have been in contact with a medium incompatible to the requested transfer medium may not be considered.
- Use material-medium compatibility matrix: Components that are made of a material incompatible to the requested transfer medium may not be considered.

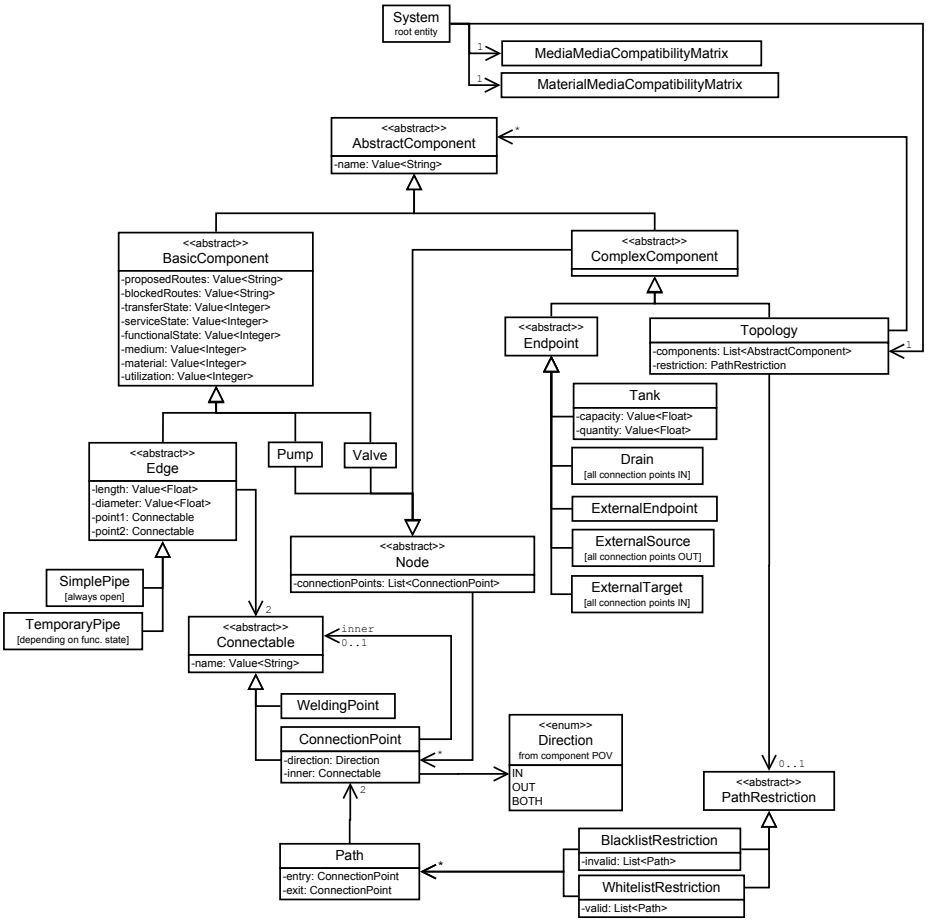

Fig. 4. Class diagram of the ontology defining the representation of the physical system components

As the route finding algorithm calculates the route from node to node, these criteria act likewise to a filter reducing the set of neighbours of each node, which can be considered for calculating the next route segment. Moreover, the application of backtracking allows more complex route finding behaviors such as the criterion of having to use a pump for the transfer. In this case, the calculated route must contain one pump component, while taking into account that the parts of the route before but also after the pump have to be determined in regard to the total transport costs.

5 Implementation of the Approach

The following sections reveal details about the demonstration applications, which prove the feasibility of the approach.

Fig. 5. The laboratory process plant (a) and a screenshot of the graphical user interface provided by the order agent (b)

5.1 Implementation on the Laboratory Process Plant

A laboratory process plant is used as the target system for implementing the approach (see Figure 5a). It encompasses common industrial components for providing training possibilities in regard to open-loop as well as closed-loop control technologies. Even though the complexity of the recipes to be executed on this laboratory process plant is rather limited, it is suitable for the implementation of a demonstration application.

The agent system is hosted and executed on a PC within the Java Agent Development Framework (Jade) [17] using its provided services such as the Directory Facilitator. The phases are provided on an industrial PLC of the type CompactLogix by Rockwell Automation [18]. For enabling the PAs (with their "agent"-part also residing in the PC) to issue state changes of their phases (see Section 2) and receive status updates, a phase logic interface in the form of Java functions is employed for writing and reading tags (i.e. variables) in the data table of the PLC. Using a simple Graphical User Interface (GUI), the operator can choose a recipe from a list of available recipes and specify the amount of liquid to be processed (see Figure 5b). The GUI offers a set of buttons (start, pause, continue, etc.) for changing the state of the momentarily processed recipe and returns according information. Due to the size of the used laboratory process plant, multiple recipes cannot be processed simultaneously. However, the equipment components and phases to be used are determined dynamically based on the operations specified in the recipe.

5.2 Evaluation of the Path Finding Algorithm on a Complex Pipe Layout

Due to the fact that the presented laboratory process plant allows only very limited routing cases and contains only basic components, the route finding

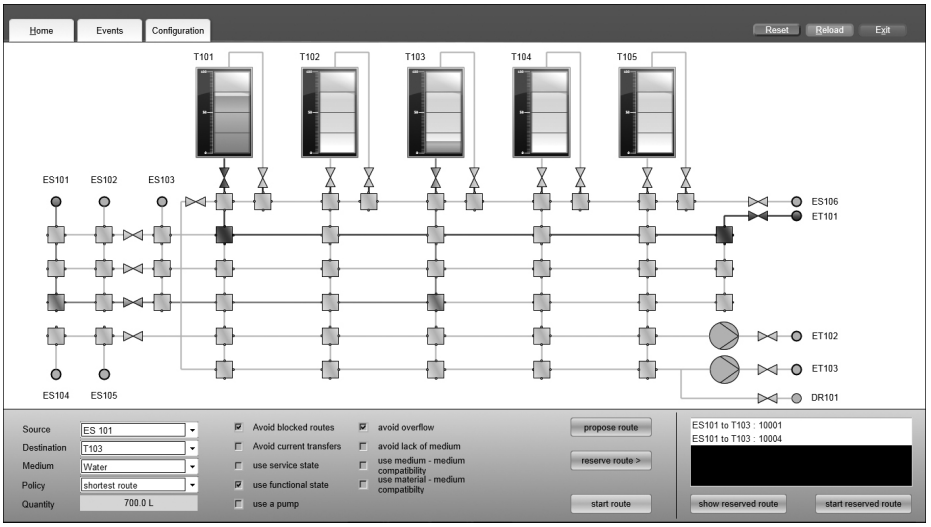

Fig. 6. Visualization in zenon with transfer tasks in execution

algorithm is tested also on a more complex pipe layout for demonstrating its benefits. An ontology representing the pipe layout is provided in an according XML file. A visualization and simulation of this pipe layout is implemented using the Supervisory Control and Data Acquisition (SCADA) software zenon [19].

As can be seen in Figure 6, the visualization incorporates a set of five tanks and external sources (ES101-ES106) as well as external targets (ET101-ET103 and DR101). The rectangular blocks in the screenshot represent a specific type of valves (realized as topologies with sub-components), which are employed to connect the horizontal with the vertical pipe lines. Transfer tasks between the tanks and the external sources and targets can be created taking also the criteria presented in Section 4 into account.

The communication between zenon and the routing algorithm is realized on the basis of an OPC Unified Architecture (OPC UA) interface [20] with a set of variables on the OPC UA server to which both zenon and the route finding algorithm have access. After pressing the button “propose route”, the route information (i.e. source, destination, quantity, etc.) and the status variables of the components (e.g. a component is already in use for a previously started transfer task) are read by the route finding algorithm for starting the calculation. Upon completion, the calculated route is written likewise on a corresponding set of proposal variables for the components and thereby the visualization is able to display the proposed route. Now the operator can start the transfer task at which an underlying simulation modifies the amount of liquid in the shown tanks accordingly and stops the process after transferring the desired amount of liquid.

Measurements show that a route calculation for the given pipe layout requires in average 5 to 6 seconds on a Dual-Core PC with 2 GHz and 2 GB RAM with no clear correlation to the chosen route criteria (see Figure 7). As process times

Route T101 to ET101	calculation time in milliseconds					
	1	2	3	4	5	Average
Avoid blocked (reserved) routes	5891	5500	5454	5469	5453	5553,4
Avoid current transfers	5500	5657	5407	5360	5422	5469,2
Use service state	5468	5593	5469	6141	5343	5602,8
Use functional state	5344	5328	5265	5312	5344	5318,6
Use a pump	5281	5407	5297	5297	5281	5312,6
Use medium/medium compatibility	5469	5485	5469	5406	5469	5459,6
Use material/medium compatibility	5437	5484	5734	5406	5360	5484,2
All criteria used	5672	5578	5563	5563	5562	5587,6

Fig. 7. Required time for the route calculation with single criteria activated (first seven rows) or all criteria activated (last row) in 5 performed runs (columns)

in the batch industry are commonly in the range of minutes, hours or days, the calculation time is well within acceptable boundaries.

6 Conclusion

This paper presents a multi-agent system for batch process automation based on phase agents for controlling the physical components. The phase agents are wrapped around control software based on the standard IEC 61131 for programming PLCs. Thus, the phases can be determined and executed dynamically according to the executed production recipe. Moreover, a route finding algorithm is introduced, which is capable of calculating routes for material transfers in consideration of various criteria such as components used or reserved by previously calculated routes. Both the agent system and the route finding algorithm are versatile concerning as a system modification requires just adding or removing the according PAs and a change of the ontology representing the system.

The feasibility of the multi-agent system is shown by its implementation on a laboratory process plant. Due to its limited size, the route finding algorithm is also evaluated on a more complex pipe layout in conjunction with an industrial batch management system in the framework of the SCADA software zenon.

Future research efforts will be concerned with testing the presented approach on an extended laboratory process plant, which allows the execution of more complex recipes requiring also more extensive transfer tasks. Moreover, an XML export wizard is currently in development, which allows the automatic generation of the system ontology based on a given visualization in zenon.

Acknowledgment. The authors acknowledge the financial support from the BRIDGE program by the Austrian Research Promotion Agency under contract FFG 829576.

References

1. Guo, Q.L., Zhang, M.: An agent-oriented approach to resolve scheduling optimization in intelligent manufacturing. Robotics and Computer-Integrated Manufacturing 26(1), 39–45 (2010)

2. Mes, M., van der Heijden, M., van Harten, A.: Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems. European Journal of Operational Research 181(1), 59–75 (2007)
3. Brennan, R.: Toward Real-Time distributed intelligent control: A survey of research themes and applications. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 37(5), 744–765 (2007)
4. Rajakumar, S., Arunachalam, V., Selladurai, V.: Workflow balancing strategies in parallel machine scheduling. The International Journal of Advanced Manufacturing Technology 23, 366–374 (2004)
5. Wahab, M., Stoyan, S.: A dynamic approach to measure machine and routing flexibilities of manufacturing systems. International Journal of Production Economics 113(2), 895–913 (2008)
6. Vrba, P., Mařík, V.: Capabilities of dynamic reconfiguration of Multiagent-Based industrial control systems. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 40(2), 213–223 (2010)
7. Merdan, M., Lepuschitz, W., Axinia, E.: Advanced Process Automation Using Automation Agents. In: Proceedings of the 5th International Conference on Automation, Robotics and Applications, pp. 34–39 (2011)
8. Jämsä-Jounela, S.L.: Future trends in process automation. Annual Reviews in Control 31(2), 211–220 (2007)
9. Leitao, P.: Agent-based distributed manufacturing control: A state-of-the-art survey. Engineering Applications of Artificial Intelligence 22(7), 979–991 (2009)
10. Jennings, N., Bussmann, S.: Agent-based control systems: Why are they suited to engineering complex systems? IEEE Control Systems Magazine 23(3), 61–73 (2003)
11. Metzger, M., Polaków, G.: A survey on applications of agent technology in industrial process control. IEEE Transactions on Industrial Informatics 7(4), 570–581 (2011)
12. International Electrotechnical Commission: IEC 61131 Programmable controllers – Part 3: Programming languages, Geneva (1993)
13. International Electrotechnical Commission: IEC 61512 Batch Control – Part 1: Models and Terminology, Geneva (1997)
14. Parker, M., Rawtani, J.: Practical Batch Process Management. Newnes, The Netherlands (2005)
15. Kuikka, S.: A batch process management framework: Domain-specific, design pattern and software component based approach. PhD thesis, Helsinki University of Technology, Espoo (1999)
16. Dijkstra, E.W.: A Note on Two Problems in Connexion with Graphs. Numerische Mathematik 1(1), 269–271 (1959)
17. Telecom Italia: Jade - Java Agent Development Framework (access date March 2013)
18. Rockwell Automation: CompactLogix Control Systems (access date March 2013)
19. COPA-DATA: HMI SCADA Software zenon by COPA-DATA (access date March 2013)
20. OPC Foundation: OPC Unified Architecture (access date March 2013)