Vladimír Mařík
Jose L. Martinez Lastra
Petr Skobelev (Eds.)

LNAI 8062

Industrial Applications of Holonic and Multi-Agent Systems

6th International Conference, HoloMAS 2013
Prague, Czech Republic, August 2013
Proceedings

Springer

Lecture Notes in Artificial Intelligence 8062

Subseries of Lecture Notes in Computer Science

LNAI Series Editors

LNAI Founding Series Editor

Vladimír Mařík
Jose L. Martinez Lastra
Petr Skobelev (Eds.)

Industrial Applications of Holonic and Multi-Agent Systems

6th International Conference, HoloMAS 2013
Prague, Czech Republic, August 26-28, 2013
Proceedings

Springer

Volume Editors

Vladimír Mařík
Czech Technical University
Department of Cybernetics
Technická 2, 166 27 Prague 6, Czech Republic
E-mail: marik@labe.felk.cvut.cz

Jose L. Martinez Lastra
Tampere University of Technology
Department of Production Engineering
Korkeakoulunkatu 6, 33720 Tampere, Finland
E-mail: jose.lastra@tut.fi

Petr Skobelev
Smart Solutions Ltd.
Moscowskoe shosse 17, 443013 Samara, Russia
E-mail: skobelev@kg.ru

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-40089-6 e-ISBN 978-3-642-40090-2
DOI 10.1007/978-3-642-40090-2
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013944671

CR Subject Classification (1998): I.2.11, J.6, I.6, D.2, J.7, J.1, I.2

LNCS Sublibrary: SL 7 – Artificial Intelligence

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

It is a real pleasure to state that the R&D activities around holonic and multi-agent systems for industrial applications have not faded during the last decade. On the contrary, the number of both the scientific topics and the achievements in the field has been and is growing steadily. Besides HoloMAS, which has been the pioneering event of this field, there are multiple conferences like IEEE SMC annual conferences, INDIN, ETFA, or INCOM that direct their attention to advanced industrial solutions based on intelligent agents.

This year's conference was the ninth in the sequence of HoloMAS events. The first three events (HoloMAS 2000 in Greenwich, HoloMAS 2001 in Munich, and HoloMAS 2002 in Aix-en-Provence) were organized as workshops under the umbrella of DEXA. Starting from 2003, HoloMAS received the status of an independent conference, organized biennially on odd years, still under the DEXA patronage (HoloMAS 2003 in Prague, HoloMAS 2005 in Copenhagen, HoloMAS 2007 in Regensburg, HoloMAS 2009 in Linz, and HoloMAS 2011 in Toulouse). The HoloMAS line of scientific events has successfully created a community of researchers who are active in the field. They have started cooperations in large EU projects, for example, the IP project ARUM in 2012, and additionally have jointly submitted several new project proposals since the last HoloMAS event.

The research of holonic and agent-based systems invokes stronger and flourishing interest in industry and receives increasing support from both the public sector and private institutions. We see increased interest from the IEEE System, Man, and Cybernetics Society, namely, from its Technical Committee on Distributed Intelligent Systems. Another IEEE body – Industrial Electronics Society – supports the related R&D field through its Technical Committee on Industrial Agents (http://tcia.ieee-ies.org/). Its mission is to provide a base for researchers and application practitioners to share their experiences with application of holonic and agent technologies in industrial sectors, especially in assembly and process control, planning and scheduling, and supply chain management. There are a number of impacted journals that provide space for articles dealing with industrial agents such as *IEEE Transactions on SMC: Systems, IEEE Transactions on Industrial Informatics, Journal of Production Research, Journal of Intelligent Manufacturing or JAAMAS.*

We are proud to announce that the HoloMAS 2013 conference was once again held under the technical co-sponsorship of the IEEE SMC Society.

It is our pleasure to inform you that since HoloMAS 2011, there have been 37 papers submitted, from which the Program Committee selected 25 regular papers and invited two additional papers to be included in this volume. This volume collects papers on agent-based architectures and solutions for industrial control and manufacturing (eight papers), another four papers are concentrated on simulation, verification and validation of MAS industrial solutions, two of them describing ontology-based approaches. Three papers are dedicated to the MAS solutions in intelligent transportation systems — this application area seems to have grown in importance in the last few years, because some of the results could easily be introduced into transportation practices. Another five papers are aimed at industrial case studies which are – as a rule – based on specifically designed architectures. The last five papers bring new theoretical results and application trends, especially focused on potential MAS extensions for cloud computing and the big data technologies.

Looking at the applications of holonic and agent-based systems, we have collected quite an interesting portfolio of papers aimed at hot application topics including smart grids, supply chain and logistics, manufacturing, automation, etc.

There were three invited talks specifically targeted toward HoloMAS 2013:

- Thomas Strasser et al: "Review Trends and Challenges in Smart Grids: An Automation Point of View"
- Vladimir Vittikh et al: "Actors, Holonic Enterprises, Ontologies and Multi-agent Technology"
- Armando Walter Colombo: "Service-Oriented Automation: Implementing Industrial Cyber-Physical Systems on the Shop Floor and Up"

We believe the HoloMAS 2013 conference represented another successful scientific event in the HoloMAS history and created a highly motivating environment, challenging the future research and fostering the integration of efforts in the subject field. HoloMAS conferences have always served as showcase platforms for demonstrating the holonic and agent-based manufacturing architectures and solutions. These have always offered and provided information about the state of the art in the MAS application field to specialists in neighboring research fields covered by the DEXA multi-conference event.

We are very grateful to the DEXA Association for providing us with this excellent opportunity to organize the HoloMAS 2013 conference within the DEXA event.

We would like to express many thanks to Mrs. Gabriela Wagner and Mrs. Jitka Seguin for all her organizational efforts, which were of key importance for the success of our conference.

We would like to thank IEEE SMC Society, and especially the Technical Committee on Distributed Intelligent Systems of this Society, for its technical co-sponsorship.

June 2013

Vladimír Mařík
Jose L.M. Lastra
Petr Skobelev

HoloMAS 2013

6th International Conference on

Industrial Applications of Holonic and Multi-Agent Systems

Prague, Czech Republic, August 26–28, 2013

Conference Co-chairs

Vladimír Mařík	Czech Technical University in Prague, Czech Republic
Jose Luis Martinez Lastra	Tampere University of Technology, Finland
Petr Skobelev	Smart Solutions, Russia

Program Committee

Emmanuel Adam	University of Valenciennes and Hainaut Cambresis, France
José A. Barata	Universidade Nova de Lisboa, Portugal
Theodeor Borangiu	University of Bucharest, Romania
Vicent Botti	Universitat Politècnica de València, Spain
Robert W. Brennan	University of Calgary, Canada
Armando W. Colombo	University of Applied Sciences Emden - Schneider Electric, Germany
Adriana Giret	Universidad Politecnica de Valencia, Spain
William Gruver	Simon Fraser University, Canada
Kenwood H. Hall	Rockwell Automation, USA
Udo Inden	Cologne University of Applied Sciences, Germany
Jacqueline Jarvis	Central Queensland University, Australia
Toshiya Kaihara	Kobe University, Japan
Kari Koskinen	Aalto University, Finland
Paulo Leitão	Polytechnic Institute of Braganca, Portugal
Wilfried Lepuschitz	Vienna University of Technology, Austria
Michael Luck	King's College London, UK
Bjorn Madsen	LEGO Global Distribution & Logistics, Denmark
Luis M. Camarinha-Matos	Universidade Nova de Lisboa, Portugal
Francisco Maturana	Rockwell Automation, USA
Maxim Matushin	Rocket Space Corporation "Energia", Russia
Duncan McFarlane	Cambridge University, UK
Nikolay Mehandjiev	University of Manchester, UK
Munir Merdan	Vienna University of Technology, Austria

Lars Moench	University of Hagen, Germany
Marek Obitko	Rockwell Automation, Czech Republic
Michal Pěchouček	Czech Technical University in Prague, Czech Republic
Milan Rollo	Czech Technical University in Prague, Czech Republic
Stuart Rubin	SPAWAR Systems Center, San Diego, USA
George Rzevski	The Open University, UK
Ilkka Seilonen	Aalto University, Finland
Leonid Sheremetov	Mexican Oil Institute, Mexico
Kwang Mong Sim	University of Kent, UK
David Šišlák	Czech Technical University in Prague, Czech Republic
Alexander Smirnov	SPIIRAS, Russia
Thomas Strasser	AIT Austrian Institute of Technology, Austria
Pavel Tichý	Rockwell Automation, Czech Republic
Damien Trentesaux	University of Valenciennes and Hainaut Cambresis, France
Paul Valckenaers	Catholic University Leuven, Belgium
Jiří Vokřínek	Czech Technical University in Prague, Czech Republic
Pavel Vrba	Czech Technical University in Prague, Czech Republic
Valeriy Vyatkin	Lulea University of Technology, Sweden
Alois Zoitl	fortiss, Germany

Organizing Committee

Jitka Seguin	Czech Technical University in Prague, Czech Republic
Gabriela Wagner	DEXA Society, Austria

Table of Contents

Design, Simulation and Validation

MAS in Transportation Systems

Industrial Applications

New Trends

Review of Trends and Challenges in Smart Grids: An Automation Point of View

Thomas Strasser[1], Filip Andrén[1],
Munir Merdan[1], and Alexander Prostejovsky[2]

[1] Energy Department—AIT Austrian Institute of Technology
Giefinggasse 2, 1210 Vienna, Austria
{thomas.strasser,filip.andren,munir.merdan}@ait.ac.at
http://www.ait.ac.at

[2] Automation and Control Institute (ACIN)—Vienna University of Technology
Gusshausstr. 27-29/376, 1040 Vienna, Austria
prostejovsky@acin.tuwien.ac.at
http://www.acin.tuwien.ac.at

Abstract. Low-carbon and energy efficient technologies are seen as key enablers to reduce the green-house gas emissions and to limit the global warming. The large scale penetration of distributed energy resources from renewables seems to be a very promising approach. In order to cope with the fluctuating nature of such energy resources an intelligent integration in today's electric energy infrastructure is necessary. Such intelligent power networks—called Smart Grids—tend to have a higher complexity compared to the traditional infrastructure. They need advanced control approaches in order to be manageable. The development of more sophisticated information, communication and automation technologies and control algorithms are in the focus of the research community today in order to master the higher complexity of Smart Grid systems. This paper provides a review of automation trends and challenges for the future electric energy infrastructure with focus on advanced concepts using artificial intelligence and multi-agent systems. Moreover, most important standards and common rules are also discussed in order to satisfy interoperability issues in such a distributed environment.

Keywords: Distributed Energy Resources, Information and Communication Technology, Multi-agent Systems, Power Utility Automation, Renewable Energy Sources, Supervisory Control and Data Acquisition, Smart Grids, Standards.

1 Introduction

In order to cope with the worldwide climate change low-carbon and energy efficient technologies are required. A dramatic CO_2 reduction and accompanied by limiting the gradual increase in the global average temperature—addressed by various national and regional policy and strategy roadmaps—can only be achieved by reducing the usage of fossil fuels for electricity generation [1]. The ongoing world-wide industrialization, the growth in population as well as the

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 1–12, 2013.

improved welfare and living standards and accompanying the ever increasing global energy needs make it very difficult to achieve the aforementioned challenging goals. Consequently, a change of the worldwide energy mix moving towards a smarter integration of Renewable Energy Sources (RES) from wind, solar, geothermal or biomass into today's energy networks is necessary in order to achieve the ambitious targets for CO_2 reduction and global warming [1].

One main issue in current research activities regarding power distribution networks is the operation of the grid with a high penetration of Distributed Energy Resources (DER). Particularly, the energy generation from RES is characterized by intermittent and only partially predictable production [1, 2]. In order to keep power quality and grid stability, a large effort is necessary to control the power generators and/or the energy demand (i.e., loads). A high penetration of such Distributed Generators (DG) normally leads to a more complex power distribution network as today. In order to manage such a critical infrastructure advanced management technologies are necessary [2]. Recent developments in Information and Communication Technology (ICT) as well as in industrial automation have the potential to provide advanced control methods in order to supervise, monitor and manage the future electric energy infrastructure—called Smart Grid—with a high amount of RES [3, 4].

The main aim of this paper is to provide a review of recent research trends and challenges in Smart Grids with focus on automation and control topics and to analyze open issues. Since the future electric energy infrastructure is characterized by a high amount of DGs also interoperability requirements have to be addressed. A brief overview of the most important Smart Grids related standards are also provided within this paper.

The rest of paper is organized as follows: Section 2 gives an introduction about the transition of power distribution grids into Smart Grids due to the integration of Renewable Energy Sources (RES). Also the need for a higher automation degree in distribution networks is motivated and important Smart Grid automation functions are analyzed in this section, too. An overview of advanced automation technologies for Smart Grid Systems addressing future needs as well as important Information and Communication Technology (ICT) related standards are provided in Section 3. This paper is concluded with the discussion of open issues and research trends for the further development and implementation of automation approaches in Smart Grids.

2 Towards Active Power Distribution Grids

This section provides a brief overview about the main trends in active power distribution grids (i.e., Smart Grids). A comparison of the traditional topology of the electric energy infrastructure with a Smart Grid systems motivates the need of a higher automation degree.

2.1 Integration of Renewable Energy Resources

The traditional electric energy infrastructure is characterized by a central generation infrastructure using large-scale power plants (i.e., bulk generation). By using power transmission line on high voltage levels (i.e., normally voltages levels >100kV) the centralized generated electricity is transported over long distances to regions and cities. A series of different step-down transformers are used to distribute it over medium voltage (i.e., normally voltage levels in the range of >1kV and <100kV) and low voltage distribution grids (i.e., normally voltages levels <1kV) to the customers [5–7]. Summarizing, the traditional system is characterized by a hierarchical structure and an unidirectional power flow from the central generators to the customers as depicted in the following Figure 1.

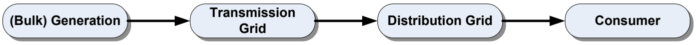


Fig. 1. Overview of the traditional electric energy infrastructure (∼1900-2000)

With the integration of DER based on renewables the topology of the electric energy infrastructure changes. Figure 2 provides an overview of a Smart Grid with centralized and distributed generation. The RES/DER generators are normally connected to the low voltage and medium voltage distribution grids causing a bidirectional power flow due to the local generation. The topology of the whole system has still a hierarchical nature but the Smart Grid has to cope with fluctuating distributed generation as motivated in the introduction. [6, 7]

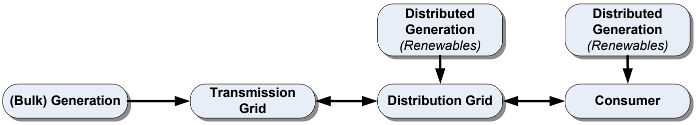


Fig. 2. Overview of the future electric energy infrastructure (from ∼2000)

Moreover, not only the DGs but also the upcoming large-scale integration of Electric Vehicles and the enhanced possibilities through the availability of controllable loads (e.g., through local thermal, mechanical and electro-chemical storages like heat pumps, batteries) makes it more challenging to manage such a Smart Grid system. [6, 7]

2.2 Towards a Higher Automation Degree in Power Grids

In the past the operation of the transmission and distributed grids and the connected components (i.e., primary and secondary substations, central generators) have been done mainly manually. Today, with the availability of powerful Supervisory Control and Data Acquisition (SCADA) systems and corresponding remote control possibilities a higher automation degree can be usually achieved in the power grids. Especially in Europe the interconnected power transmission grids are today characterized by a high degree of automation: From transmission control centers remote connections to bulk generation and corresponding substations are established in order to keep the transmission system under control. In addition, Phasor Measurement Units (PMU) are used for wide area network monitoring to keep the transmission grids under control. [2, 4]

Usually, the automation level in power distribution grids is much lower compared to the transmission system. Secondary substations and larger DGs are controlled by distribution control centers. Distribution automation is currently mainly applied in the medium voltage distribution grids whereas low voltage grids are mainly manually operated. [2, 4]

Due to the large scale integration of DER components and the future penetration of EV and decentralized storages it is expected that the automation degree will also rise in the distribution grids (i.e., medium and low voltage level). All components (i.e., generators, loads, storages, transmission and distribution lines, substations, on-load tap change transformers, measurement devices, smart meters) in the future power grid (i.e., Smart Grid) are interconnected in addition to the electric infrastructure with a powerful communication network and corresponding automation system in order to monitor, manage and optimize it in a more intelligent manner [2].

2.3 Necessary Automation Functions for Future Grids

In order to control and operate future grids the following important automation functions have to be covered by the corresponding automation and control system. The following list provides a brief overview of the most important functions [2, 4, 5, 7–9]:

- *Self-healing:* Automatic restoration of grid operation in case of faults/errors
- *Self-optimization:* Ability to optimize the grid operation due to fluctuating generation from RES and the availablitly of controllable loads/storages
- *Self-monitoring and -diagnostics:* Advanced monitoring and state estimation capability; real-time or near real-time based condition monitoring of the grid components and devices
- *Condition dependent maintenance:* Preventive maintenance according to component condition and remaining life-time
- *Automatic grid (topology) reconfiguration:* Automatic adjustment of the grid topology for grid optimization (e.g., max. amount of DGs and EVs) or fault management and system restoration
- *Adaptive protection:* Automatic adaption of protection equipment settings due to actual grid condition (e.g., adaptation of the protection system settings due to the bidirectional power flow)
- *Demand response support:* Advanced energy management taking distributed generation and controllable loads/storages into account
- *Distributed management:* Distributed management and control with automatic decision finding process and proactive fault/error prevention

- *Distributed generators with ancillary services:* Possibility to use ancillary services (e.g., voltage/frequency control) of DERs for grid optimization
- *Advanced forecasting support:* Forecasting of generation and load profiles for optimized grid operation

The state-of-the-art technology (i.e., centralized Transmission/Distribution Management Systems, SCADA systems) is not able to take all the above mentioned Smart Grid automation functions into account. More advanced and sophisticated automation technologies and methods are necessary in order to control the future Smart Grids. Key requirements for the ICT and automation systems are related to flexibility, adaptability, scalability and open interfaces. [9]

3 Automation Technologies and Standards for Smart Grids

Due to the large scale integration of DERs, storages and EVs in the future Smart Grids, as motivated above, advanced automation methods and related automation functions are necessary. This section provides an overview of Multi-agent Systems and artificial intelligence technology for the automation and control of Smart Grid systems. Moreover, an analysis of the most important ICT standards are provided addressing mainly interoperability requirements in the future grids.

3.1 Multi-Agent Systems Based Smart Grid Control

Multi-agent Systems (MAS) are considered as an approach able to realize/implement the above described functionalities and services for future Smart Grids including intelligent devices/ components [9, 10]. In general, MAS technology offers a convenient way to cope with the dynamics in large complex systems, making the control of the system decentralized, thereby reducing the complexity, and increasing flexibility, as well as enhancing fault tolerance [11] as motivated in the previous section. MAS are composed of distributed heterogeneous units/agents, where each agent manages its own activities on the basis of its local state and the information (messages) received from other agents. Besides its usage in industrial environments like manufacturing and logistics, agent-based approaches are also suitable for power and energy systems. For example, an agent can supervise one or several physical components of the electric energy infrastructure (e.g., substation/transformer, Smart Meter, generators, switches, loads) or functional entities (e.g., forecasting job, energy management, diagnostics). A distributed system architecture enables local data processing and minimizes the need for massive data exchanges (e.g., capable of providing local fail proof, feeder level forecasts aggregated at substation level). A distributed system can enable the high performance needed for preventing or containing rapidly evolving adverse events [12]. Figure 3 shows an example applying MAS technology in power grids.

In electric power systems agents are used for diverse complex services, e.g., for controlling, communication and protection of the grid [13, 14], monitoring,

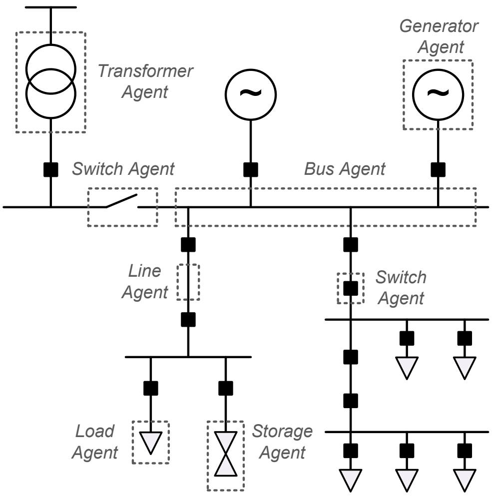


Fig. 3. Example for applying MAS technology in Smart Grids

visualizing and coordinating energy use within smart homes [15, 16], analyzing and determining states and market situations [17, 18], as well as scheduling and forecasting of load consumption and power generation [19]. Several authors have implemented a Multi-agent System to control the operation of a microgrid [20, 21]. Some other authors have addressed the issue of fault diagnosis and reconfiguration [22, 23]. In this context, Li Liu et al. [24] have described the need for fault detection in naval shipboard power systems and Huang et al. [25] have presented a multi-agent approach for fault detection using nonlinear parameter identification techniques. Solanki et al. [10, 26], have presented different approaches for restoration of the power system by distributed reconfiguration. Chouhan et al. [27] as well as Merdan et al. [7] have proposed a multi-agent based system for distribution network reconfiguration for optimization purposes.

3.2 Artificial Intelligence Based Control Approaches

Besides the MAS method artificial intelligence based approaches are also suitable for Smart Grid operation. For example, the Particle Swarm Optimization (PSO) technique is a comparable method to MAS. This method—a metaheuristic—is based on swarm intelligence and normally all swarm members tend to converge towards the optimal solution. Similar to MAS also PSO can be used for the topology reconfiguration and optimization of power distribution grids [28]. Moreover, PSO approaches are also very suitable for power grid planning [29, 30].

In addition, Artificial Neural Networks (ANN) can also be used for complex optimization, planing and scheduling tasks. ANNs are based on biology-based concepts and they are oriented on the information architecture of the brain and nervous system of animals and humans. They consist basically of neurons and therefore their connections. Usually, there are two different phases in the usage of ANN. First, the ANN must be trained to a certain behavior before they can be used. In the literature ANN are also used for complex planning and optimization tasks in the power and energy domain. For example, power grid topology reconfiguration tasks have been already solved using ANN [31, 32].

3.3 Important ICT-Based Standards

In the dynamically developing field of ICT solutions for Smart Grids the development of standards is of crucial importance mainly due to interoperability requirements. Currently, standards are mainly focused on non-distributed devices and systems since they originated prior to the increasing deployment of distributed generators. In order to develop an interoperable solution, international standards have to be taken into account. Therefore, it is very important to keep the development of Smart Grids components aligned with standards and documented methods that describe and chart how their development should be accomplished. Among other things this involves standards on how to implement the components and how they should communicate with each other. Several standardization organizations and various international projects have analyzed this fact so far. For example, Specht et al. [33] and Gungor et al. [34] provide a comprehensive overview of important Smart Grids standards and international roadmaps. In the following paragraphs a brief of the most important international activities and standards for Smart Grids systems is provided.

On international level the International Electrotechnical Commission (IEC) plays a very important role to provide common rules and standards for the planning and operation of active power distribution grids. Especially, the IEC Technical Committees (TC) TC 8 ("Systems aspects for electrical energy supply"), TC 13 ("Electrical energy measurement, tariff- and load control "), TC 57 ("Power systems management and associated information exchange"), and TC 65 (" Industrial-process measurement, control and automation") are responsible for Smart Grid related standards as mentioned in the "IEC Smart Grid Standardization Roadmap" [35]. In this report the IEC suggests the following core standards to be used for the realization of Smart Grid related projects [35]:

- *IEC TR*[1] *62357:* Proposes a service-oriented reference architecture for Energy Management Systems (EMS) in the power transmission domain and for Distribution Management Systems (DMS) in the power distribution domain
- *IEC 61970/61968:* Introduces a domain ontology called Common Information Model (CIM) for modeling the electrical grid and its components
- *IEC 61850:* Covers the automation of substations and power utility equipment (e.g., DER components)
- *IEC 62351:* Describes security issues
- *IEC 62056:* Provides data exchange rules and specifications for meter reading, tariff and load control in power systems
- *IEC 61508:* Addresses functional safety rules for electrical, electronic and programmable electronic safety-related systems and devices

[1] Technical Report.

In addition the following IEC standards are also ranked with high relevance for Smart Grid Systems development [35]:

– *IEC 60870-5:* Covers tele-/remote control protocols
– *IEC 60870-6:* Responsible for inter control center communication
– *IEC TR 61334:* Addresses Distribution Line Message Service (DLMS)
– *IEC 61400-25:* Defines wind power communication rules
– *IEC 61850-7-410:* Covers hydro energy communication issues
– *IEC 61850-7-420:* Covers distribution Energy communication issues
– *IEC 61851:* Covers Electric Vehicle (EV) communication, Smart Home, and E-mobility
– *IEC 62051-54/58-59:* Defines metering-related standards

Similar results and suggestions have also been reported by other important Smart Grids related roadmaps like the "NIST Framework and Roadmap for Smart Grids Interoperability Standards" [36], the "DKE German standardization roadmap for Smart Grids" [37], as well as the IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads [38], to mention the most important once.

Moreover, the German DKE Smart Grids roadmap suggests for the the implementation of control logic the IEC TC 65 approaches IEC 61131 for Programmable Logic Controllers (PLC) or IEC 61499 for distributed intelligent automation systems. They provide promising concepts for the development of control solutions in the Smart Grids domain [37]. In addition, a further IEC TC 65 automation-oriented approach (i.e., IEC 62541), the OPC Unified Architecture (OPC UA), gets a lot of interest from the Smart Grid community [39]. Since OPC Data Access (OPC DA) was especially developed for Windows platforms, the new OPC UA specification provides a platform-independent information and communication model based on SOA for the data exchange in industrial automation systems. There are several ongoing activities to use the object-oriented OPC-UA data model and corresponding communication protocols to map the CIM and the IEC 61850 models to it [40, 41].

Another very interesting standard-related activity is the "M/490 Standardization Mandate to European Standardization Organizations (ESO)" of the European Commission to support European Smart Grid deployment [8]. The goal of this work performed by the ESO's is to develop a so-called "Smart Grid Architecture Model (SGAM)" which covers different concepts and viewpoints as well as a mapping methods for a structured approach to develop interoperable Smart Grid solutions. It combines the physical/components layer with all elements of the energy conversation chain and different interoperability categories (i.e., communication, information, function, business) as defined by the Grid Wise Architecture Council (GWAC) [42] to a three-dimensional representation.

4 Open Issues and Research Trends

The integration of new components and devices (e.g., DERs, EVs, storages) into the power grids urge for a higher automation degree in order to operate such systems with a higher complexity. Recent development in the ICT domain can provide suitable approaches. Within this paper an analysis of important automation functions for the operation of future Smart Grids has been carried out. Moreover, a review of advanced automation technologies and approaches has shown that concepts motivated from biology like MAS, PSO and ANN have the potential to be used to build the basis for a Smart Grid automation and control system. However, despite the growing awareness and the ongoing research activities in the field of power distribution automation approaches, there are still several open issues that have to be addressed in future research work such as:

- *Advanced control functions:* The ability to provide self-corrective reconfiguration and restoration (incl. hardware and software adaption) and to handle the fluctuating behavior of DER components as well as the upcoming integration of EVs is still an open point for further research. An integrated solution addressing advanced functions as mentioned in Section 2.3 is missing today. Such an integrated approach should be based on MAS/SOA-based concept in order to guarantee interoperability, flexibility and scalability.
- *Universal Grids/Hybrid Grids:* A lot of efforts are spent on the research and further development of advanced functions for the electric energy infrastructure. A more comprehensive view on the whole energy systems incl. other domains (e.g., gas, district heating, water system) is a further and necessary step [43] in order to achieve the vision of a green energy supply. Approaches, concepts, methods and standards form the electricity domain should be analyzed and applied to other energy-related domains [44].
- *Formal and model-driven design of automation systems:* A common modeling language of automation applications and a model-driven design support system is required in order to support the development process of automation solutions in the Smart Grid domain [45, 46]. Such an approach would increase the design and implementation productivity and would also reduce implementation errors. It should also be able to design MAS-based approaches.
- *Advanced validation methods:* Since field tests are hardly achievable in electricity systems, proper validation methods have to be developed. In order to test and validate advanced automation functions and corresponding systems, a co-simulation approach seems to be a suitable approach covering the electrical systems, the communication network and the control algorithms. Some parts of a complex Smart Grid system can also be tested in a hardware-in-the-loop laboratory setup with more realistic behavior. The availability of suitable models (physical system, communication protocols, automation functions) as well as the coupling of the simulators is still a broad field for further research. Also the real-time simulation of the different component models for hardware-in-the-loop setups is still an unsolved issue.
- *Interoperability, scalability and harmonization of standards:* Developments in the area of advanced automation solutions based on MAS, PSO, ANN, etc. for Smart Grids have to be aligned with domain standards (motivated in Section 3.3. The usage of the SGAM and the IEC Smart Grids related standards are a good starting point but existing and future standards have

to be harmonized. They have to be used also for the development of advanced automation functions and optimization approaches like MAS, PSO and ANN. For example, standards like the IEC 61850 or the OPC UA should be used for agent-based communication as suggested by Merdan et al. [7].

– *Privacy and security:* A further advance in the degree of automation of power grids will increase the amount of communication and require a highly developed communication. Moreover, the interaction between different approaches like Building2Grid, Consumer2Grid and Vehicle2Grid will also enhance the needs of suitable communication approaches, security and privacy. To satisfy the primary or secondary control within the power grid, the reaction times have to be as small as possible and the link should be confidential [47].

Acknowledgments. This work is partially funded by the Austrian Climate and Energy Fund with the support of the Austrian Research Promotion Agency (FFG) under the project *"DG-EV-HIL"* (No. 827987). Moreover, funding is also provided by the Austrian Research Promotion Agency (FFG) under the Research Studios Austria program for the project *"MASGrid"* (No. 832037).

References

1. IEA: World Energy Outlook 2012. Technical report, Int. Energy Ag., IEA (2012)
2. IEA: Smart grid insights. Technical report, Int. Energy Agency, IEA (2011)
3. Amin, S., Wollenberg, B.: Toward a smart grid: power delivery for the 21st century. IEEE Power and Energy Magazine 3(5), 34–41 (2005)
4. Farhangi, H.: The path of the smart grid. IEEE Power and Energy Magazine 8(1), 18–28 (2010)
5. European Commission, et al.: European SmartGrids technology platform: vision and strategy for Europe's electricity networks of the future. In: Directorate-General for Research-Sustainable Energy Systems (2006)
6. Bouhouras, A., Andreou, G., Labridis, D., Bakirtzis, A.: Selective Automation Upgrade in Distribution Networks Towards a Smarter Grid. IEEE Transactions on Smart Grid 1(3), 278–285 (2010)
7. Merdan, M., Lepuschitz, W., Strasser, T., Andrén, F.: Multi-Agent system for self-optimizing power distribution grids. In: 2011 5th International Conference on Automation, Robotics and Applications, ICARA (2011)
8. EC: M/490 Standardization Mandate to European Standardisation Organisations (ESOs) to support European Smart Grid deployment. Technical report, European Commission (2012)
9. McArthur, S., Davidson, E., Catterson, V., Dimeas, A., Hatziargyriou, N., Ponci, F., Funabashi, T.: Multi-Agent Systems for Power Engineering Applications—Part I: Concepts, Approaches, and Technical Challenges. IEEE Transactions on Power Systems 22(4), 1743–1752 (2007)
10. Solanki, J., Khushalani, S., Schulz, N.: A Multi-Agent Solution to Distribution Systems Restoration. IEEE Trans. on Power Systems 22(3), 1026–1034 (2007)
11. Jennings, N., Bussmann, S.: Agent-based control systems: Why are they suited to engineering complex systems? IEEE Control Systems 23(3), 61–73 (2003)
12. Moslehi, K., Kumar, R.: A reliability perspective of the smart grid. IEEE Transactions on Smart Grid 1(1), 57–64 (2010)

13. Jung, J., Liu, C.C.: Multi-agent system technologies and an application for power system vulnerability. In: 2003 IEEE Power Eng. Society General Meeting (2003)
14. Tsai, M.S., Piclova, P., Wu, W.C., Chan, C.K.: Development of a novel multi-agent based self-healing distribution systems. In: Proceedings of the Fourth IASTED International Conference. Number 606/037 (2008)
15. Cristaldi, L., Monti, A., Ottoboni, R., Ponci, F.: Multiagent based power systems monitoring platform: a prototype. In: 2003 IEEE Power Tech Conference (2003)
16. Vytelingum, P., Voice, T.D., Ramchurn, S.D., Rogers, A., Jennings, N.R.: Intelligent agents for the smart grid. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, vol. 1. International Foundation for Autonomous Agents and Multiagent Systems (2010)
17. Dimeas, A., Hatziargyriou, N.: Design of a MAS for an Island System. In: 2007 Int. Conf. on Intelligent Systems Applications to Power Systems (2007)
18. Nguyen, P., Kling, W., Myrzik, J.M.A.: Promising concepts and technologies for future power delivery systems. In: 2007 42nd International Conference on Universities Power Engineering (2007)
19. Rehtanz, C.: Autonomous systems and intelligent agents in power system control and operation. Springer (2003)
20. Dimeas, A., Hatziargyriou, N.: A MAS architecture for microgrids control. In: Proceedings of the 13th International Conference on Intelligent Systems Application to Power Systems (2005)
21. Phillips, L.R., Weiland, L., Smith, R.B., Link, H.E.: Agent-based control of distributed infrastructure resources (2006)
22. Hossack, J., McArthur, S.D.J., McDonald, J., Stokoe, J., Cumming, T.: A multi-agent approach to power system disturbance diagnosis. In: Fifth Int. Conference on Power System Management and Control (Conf. Publ. No. 488) (2002)
23. McArthur, S.D.J., Davidson, E., Hossack, J., McDonald, J.: Automating power system fault diagnosis through multi-agent system technology. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences (2004)
24. Liu, L., Logan, K.P., Cartes, D., Srivastava, S.: Fault Detection, Diagnostics, and Prognostics: Software Agent Solutions. IEEE Transactions on Vehicular Technology 56(4), 1613–1622 (2007)
25. Huang, K., Srivastava, S., Cartes, D., Liu, L.: Agent Solutions for Navy Shipboard Power Systems. In: IEEE International Conference on System of Systems Engineering, SoSE 2007 (2007)
26. Solanki, J., Schulz, N., Gao, W.: Reconfiguration for restoration of power systems using a multi-agent system. In: Proceedings of the 37th Annual North American Power Symposium (2005)
27. Chouhan, S., Wan, H., Lai, H., Feliachi, A., Choudhry, M.: Intelligent reconfiguration of smart distribution network using multi-agent technology. In: IEEE Power Energy Society General Meeting, PES 2009 (2009)
28. Qing Sheng, S., Cao, Y., Yao, Y.: Distribution Network Reconfiguration Based on Particle Swarm Optimization and Chaos Searching. In: Asia-Pacific Power and Energy Engineering Conference, APPEEC 2009 (2009)
29. Ganguly, S., Sahoo, N., Das, D.: Multi-objective planning of electrical distribution systems using particle swarm optimization. In: International Conference on Electric Power and Energy Conversion Systems, EPECS 2009 (2009)
30. Ren, P., Li, N., Gao, L.: Optimal planning of high-voltage transmission network using the chaotic particle swarm optimization. In: 2010 Chinese Control and Decision Conference, CCDC (2010)

31. Moustafa, Y., Amer, A.H., Mansour, M., Temraz, H., Madeour, M.: An artificial neural network for optimum topology in distribution expansion planning. In: Canadian Conference on Electrical and Computer Engineering (1996)
32. Xianbo, K., Weixin, G.: A New Algorithm for Distribution Network Planning. In: Int. Conf. on Power System Technology, PowerCon 2006 (2006)
33. Specht, M., Rohjans, S., Trefke, J., Uslar, M., Gonzâlez, J.M.: International Smart Grid Roadmaps and their Assessment. EAI Endorsed Transactions on Energy Web 13(1-6) (2013)
34. Gungor, V., Sahin, D., Kocak, T., Ergut, S., Buccella, C., Cecati, C., Hancke, G.: Smart Grid Technologies: Communication Technologies and Standards. IEEE Transactions on Industrial Informatics 7(4), 529–539 (2011)
35. SMB Smart Grid Strategic Group (SG3): IEC Smart Grid Standardization Roadmap. Technical Report Ed. 1.0, International Electrotechnical Commission (IEC), Geneva, Switzerland (June 2010)
36. NIST: NIST Framework and Roadmap for Smart Grid Interoperability Standards. Technical Report NIST Special Publication 1108, National Institute of Standards and Technology - U.S. Department of Commerce, USA (2010)
37. DKE: The German Standardisation Roadmap E-Energy/Smart Grid. Technical report, German Commission for Electrical, Electronic & Information Technologies of DIN and VDE, Frankfurt, Germany (2010)
38. IEEE: IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads. Technical Report 2030-2011, Institute of Electrical and Electronics Engineers (IEEE), New York, USA (October 2011)
39. Mahnke, W., Leitner, S.H., Damm, M.: OPC unified architecture. Springer (2009)
40. Rohjans, S., Uslar, M., Appelrath, H.: OPC UA and CIM: Semantics for the smart grid. In: 2010 IEEE PES Trans. and Distr. Conference and Exposition (2010)
41. Sucic, S., Bony, B., Guise, L., Jammes, F., Marusic, A.: Integrating DPWS and OPC UA device-level SOA features into IEC 61850 applications. In: IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society (2012)
42. GWAC: GridWise Interoperability Context—Setting Framework. Technical report, The GridWise Architecture Council, GWAC (2008)
43. Gawlik, W.: From Smart Grid to Universal Grid–Generation, Storage and Communication in Hybrid Networks. In: 3rd ComForEn (2012)
44. Uslar, M., Andrén, F., Mahnke, W., Rohjans, S., Stifter, M., Strasser, T.: Hybrid grids: ICT-based integration of electric power and gas grids - A standards perspective. In: 2012 3rd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies, ISGT Europe (2012)
45. Bredillet, P., Lambert, E., Schultz, E.: CIM, 61850, COSEM standards used in a model driven integration approach to build the smart grid service oriented architecture. In: 2010 First IEEE International Conference on Smart Grid Communications (SmartGridComm), pp. 467–471. IEEE (October 2010)
46. Andrén, F., Stifter, M., Strasser, T.: Towards a Semantic Driven Framework for Smart Grid Applications: Model-Driven Development Using CIM, IEC 61850 and IEC 61499. Informatik-Spektrum, 1–11 (2012)
47. Kienesberger, G., Meisel, M., Adegbite, A.: A comprehensive information platform for the smart grid. In: AFRICON (2011)

Actors, Holonic Enterprises, Ontologies and Multi-agent Technology

Vladimir A. Vittikh, Vladimir B. Larukhin, and Alexander V. Tsarev

Institute of Control of Complex Systems of Russian Academy of Science, Russian Federation
443020 Samara, Sadovaya str. 61
vittikh@iccs.ru
Smart Solutions, Ltd., Russian Federation 443013 Samara, Moskovskoye shosse 17,
Business center "Vertical", office 1201
at@smartsolutions-123.ru

Abstract. Holonic enterprises are considered as one of the new types of networking organizations used to increase business efficiency. But in practice it requires to introduce a new type of management stimulating the interpreneur approach, active interaction between actors within the project teams, coordinated and effective decisions on resource scheduling in case of unpredictable events (new projects, breakdowns, delays, illnesses etc.). In the paper we propose the concept of holonic enterprise based on the principles of intersubjective management and describe Smart Enterprise system for supporting decision making for such types of enterprises. The system combines top-down project planning with real time on-line and bottom-up communication with employees. This provides more intelligent, flexible and efficient resource scheduling in case of unexpected events. The functionality and architecture of Smart Enterprise system based on the use of ontology and multi-agent technology is outlined. The first results of the system use and perspectives are analyzed.

Keywords: actor, intersubjectiveness, holon, management, ontology, project, multi-agent technology, resource scheduling, real time.

1 Introduction

The challenges of the global economics associated with the growing complexity of business, a priory uncertainty and high dynamics of supply and demand changes make businesses look for new approaches to increase the efficiency of the modern project-based enterprises. Modern science of management considers bureaucracy as a main obstacle for efficiency of organization, in particularity, because of hierarchic reporting with the losing content of tasks and ignoring personal knowledge and talents of employees [1-2].

The solution of the problem is considered in changing paradigm of management by shifting from centralized, monolithic, hierarchical top-down structures to talent-driven networking organizations based on the principles of intersubjective communication and negotiations between actors [3]. However, to implement such approach in

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 13–24, 2013.

practice requires intelligent systems that can operate in real time and constantly communicate with employees to work out and coordinate the decisions on resource allocations.

The paper presents new approach of designing holonic organizations and describes Smart Enterprise system designed to support the new principles. In the first section of paper the analysis of the modern approaches to enterprise management is given and main features of the holonic enterprises are considered. In the second section we consider an example of the structure of holonic enterprise, in which the competing business centers and knowledge centers are described. Virtual project shares are considered as a method of employees motivation. In the third sectional the functionality and architecture of Smart Enterprise intelligent resource management system is presented which is designed with the use of resource-demand networks, ontologies and multi-agent technology. Finally, in the fourth section the first experience for the system is considered and recommendations for its development are given.

2 New Approach to Enterprise Management

The crisis of the modern management is already fully recognized by practioneers of innovations, as it was stated on Moscow Forum of innovations 2012 [4]. One of the sections of Moscow International Forum “Open Innovations – 2012” was fully dedicated to “interpreneurship”, since “many modern companies oriented to the success and achieving long-term results begin to develop internal culture of entrepreneurship stimulating the appearance of the entrepreneur spirit in usual daily routines, integration of personal entrepreneur skills and enterprise resources”.

This approach suggests the stimulation and encouragement of new ideas, eliminating the boundaries on scope of activities for departments and their employees (breaking the barriers), recognizing the role of knowledge in the project management, tolerance to trial and error, even faults, providing the required resources, teamwork, orientation to the result, developing the award systems for achieving best results, which, of course, requires the support of the top management. The implementation of this approach breaks the existing bureaucratic stereotypes in the company management and results in designing networking organizations and implementing the principles of organizational democracy which could be based on ideas of holonic enterprises [5].

The structure of such enterprise can be explained as a multi-level network of business centers that can self-organize on virtual market of the mother company.

The main features of such enterprises are given in Table 1.

Table 1. The characteristics of holonic enterprises

Traditional enterprise	**Holonic enterprise**
Centralization of functions	Decentralizion of functions
Hierarchical structure, rigid relations between employees	Networking structure, flexible relations between employees
Closed to environment	Open to environment

Table 1. (*continued*)

Traditional enterprise	Holonic enterprise
The knowledge in decision making is rigidly fixed, the decisions are made according to formal rules of business processes	The knowledge is not fixed, priority is in generating new knowledge, the decisions are made not formally and are situation-driven
Planning approach, all resources are allocated in advance	Market approach, the resources are allocated when necessary
Resource allocation is static and pre-defined	Resource allocation is dynamic and based on growing competencies of employees
Top-down communication according with hierarchy	Peer-to-peer communication as required for problem solving
Batch planning, following the regulations and instructions	Flexible planning in real time, following events
High level of certainty	Full uncertainty
Communications are regulated by business processes	Communications are not regulated and are situation-driven
Total control	Self-motivation
Fixed salary	Dynamic salary

3 Model of Holonic Enterprise

3.1 The Structure of Holonic Enterprise

Let's consider the structure of our company Smart Solutions as an example of holonic enterprise.

The key elements of holonic enterprise structure are business centers (BC) that work as autonomous virtual companies operating in the internal market of the mother company (MC) for projects implementation. The important feature of the structure is its internal decentralized nature based on BCs supported by the infrastructure provided by KCs, but able to operate independently.

The structure of the considered company includes the BCs shown on Fig. 1, each responsible for the projects in the corresponding domains: smart aerospace, smart factory, smart trucks, smart mobile services, smart railways, smart supply chain networks.

Knowledge centers (KC) (the name reflects knowledge as the main company resource) act as resource pools for projects implementation. They become a "home" for professional community of employees, where the level of qualification of employees is determined, competencies profiles are designed, salary is assigned and individual plan of competencies development is created.

The KCs of the company include the following:

- Analytics Center (AC) including analysts working on the proposals for government and commercial customers, gathering and specifying requirements, responsible for sales, marketing, advertising and development of contacts;
- Development Center (DC) including the platform developers, solution architects, programmers, testers and technical writers;

– Project Management Center (PMC) including business centers' leaders, project managers and coordinators, it is the operational "think tank" of the company that monitors the whole project development cycle (from idea – to implementation);
– Administration Center (AdmC) including lawyer, CFO, chief accountant, economist, office manager, IT managers and other employees.

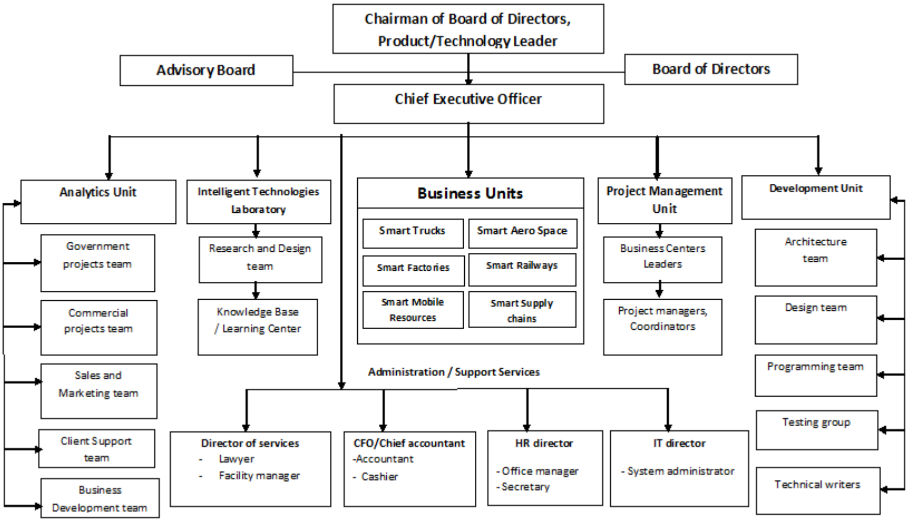


Fig. 1. Example of organizational structure of the holonic company

The role of the strategic "think tank for long-sustainable development of the company is a responsibility of the Board of Directors, which includes the leaders of BC, AC, DC, PMC, and AdmC and external advisors.

3.2 Operations of Holonic Enterprise

New BC is created on the base of the first project in the new domain sector. From the very beginning it has its own profit and loss reports and usually requires an initial investment from the mother company.

BCs directly interact with the customers and grow from project to project, by gathering domain knowledge and new opportunities. For this reason, each BC is buying the specialists from KCs as well as technology/process knowledge and ready-to-use software components and document templates, best practices etc. In its turn, KC offers and provides the employees to BCs, certifies them and monitors their growth and success in projects.

By design of the organization there exists a conflict between BCs and KCs that requires negotiations for its resolution in each specific case considering the current company situation: each BC having specific revenue is interested in obtaining the most effective specialists, but only for the period that is really required for the tasks execution, to maximize its profit. However, while the main goal of BC is the profit from projects, the goal of KC is the profit from delivery of the resources. This means that the employees, for which KCs are the "homes", must be properly trained and have the required qualification, otherwise they will not be invited by BCs.

In case of such organization each employee has the opportunity to give or receive offer his or her service to any BC negotiating on the jobs and work hours for each project. As a result of the agreements, the payments for employees becomes more and more variable and can grow unlimitedly. Employee's "home" is also interested in this since it receives the interest for each employee that participates in BC projects.

For the long-term motivation of the employees to reach to final results, we have designed the mechanism of the virtual project shares (VPS) for the employees that reflect the personal creative input into each project and results achieved.

In the case of successful task execution, the employee receives a specific amount of VPS in the projects. This is supplemented by the accumulation of the statistics on execution time for the tasks of this type which is used for new tasks estimations and generating recommendations. Finally each project has a specific sum of shares, which become a basis for bonus allocation at completing the stages of each project. For example, the employee that has successfully organized an exhibition that resulted in signing a few contracts, will receive certain "knowledge dividends" for each of these contracts.

4 Smart Enterprise Solution: Ontologies and Multi-agent Technology for Managing Holonic Enterprises

4.1 Main Functionality of Smart Enterprise Solution

There is a need for a software solution to support the full cycle of project management, covering the resource allocation, scheduling and optimization, rescheduling according to events, coordination with employees and monitoring tasks execution in real time. The solution should also take into account the semantic specifics of the tasks and how they match with the competencies of employees, support employees' interaction during decision making in real time project management, motivate people to achieve final result.

For this purpose the Smart Enterprise solution was developed on the base of resource-and-demand networks (RDN) concept using ontologies and multi-agent approach providing the real-time resource scheduling [6-7].

4.2 Multi-agent Technology for Real Time Scheduling of Holonic Enterprise

The multi-agent technologies bring the support of real-time decision making in project management by allowing the adaptive rescheduling of the resources when new tasks and other unexpected events come from outside or appear as a result of team members interaction.

Adaptiveness means that the changes in input data (events – new project, employees vacation, new tasks, changes in deadlines, execution progress updates, etc.) do not lead to the full rebuilt of the schedule, but local rescheduling is done only for the part of the schedule that is affected by these changes with possible shifting, reallocating or swapping of the operations to other employees.

The principles of multi-agent scheduling platform can be described as follows [8]:

- Each task and employee has its own software agent that receives the requirements, preferences and constraints for the scheduling and has an individual schedule;
- The agent begins the scheduling by searching for the required resources in the scene, which reflects the current situation in the department, particularly, which employee executes which tasks for the current scheduling horizon;
- If the proper employees are busy, the conflict is detected and the negotiations on its resolution by shifting, reallocating and releasing time slots begin;
- During the negotiations different options are possible: new task will be moved to the less proper resource, the previous task will leave the schedule or shift;
- Even after having solved all conflicts and made the schedule, each agent doesn't stop and pro-actively continues to look for a better result.

Such approach distinguishes the proposed system from the existing project management systems, in which all tasks and resources are considered as known in advance and do not change during scheduling or execution [9-10]. The key logic of the system is implemented by the agents that act on behalf of the orders, projects, tasks, departments, employees, software documents, etc. (Table 2). Basic flows of agents negotiations are shown in Fig. 2.

Table 2. Main agent classes of the system

Agent of	Agent description	Attributes
Order	Order looks for the best possibilities to be fulfilled within the frames of existing or new BC and KC, interests and competencies of the employees.	Contents, cost, time preferences, etc.
Project	Tries to organize and execute the project according to the given criteria, preferences and constraints, technological and business processes, availability of the employees	Ontological descriptor of the project, employees, belonging to BC, budget and deadlines, results
Organization (BC or KC, project team)	Tries to achieve and improve the results of the group in general by the given criteria, monitors the situation, changes the strategy for the selected agents, stress or destress the constraints and preferences to find out and resolve "bottlenecks".	Organization type, list of employees, criteria and strategies, expected results and current KPIs
Employee	Wants to be fully occupied according to his or her profile and receive bonuses for quality, productivity, etc. Also tries to master his or her competencies to achieve the higher level of qualification and salary.	Competencies profile, work schedule, current task, qualification level, VPS, wages, author of documents and software components, etc.
Software component or document	Wants to be maximally used in the projects, if necessary, be improved. Considers the relation between other components, documents, tests, etc.	Ontological descriptor, author, relation to other components, cost.

Table 2. (*continued*)

Technological or business process	Wants to be executed in the best way as a chain of separate operations (tasks) required to fulfill orders for the projects.	Product components, operations list and relations between them, execution criteria, cost and other terms.
Operation	Looks for the best employees, documents and components according to the preferences and constraints of the project and relations with other operations.	Competencies and qualifications of the employee, duration, relation to the project and other operations.
Result (product)	Tries to be created as a result of project execution from ready-to-use or new components	Product (results) characteristics

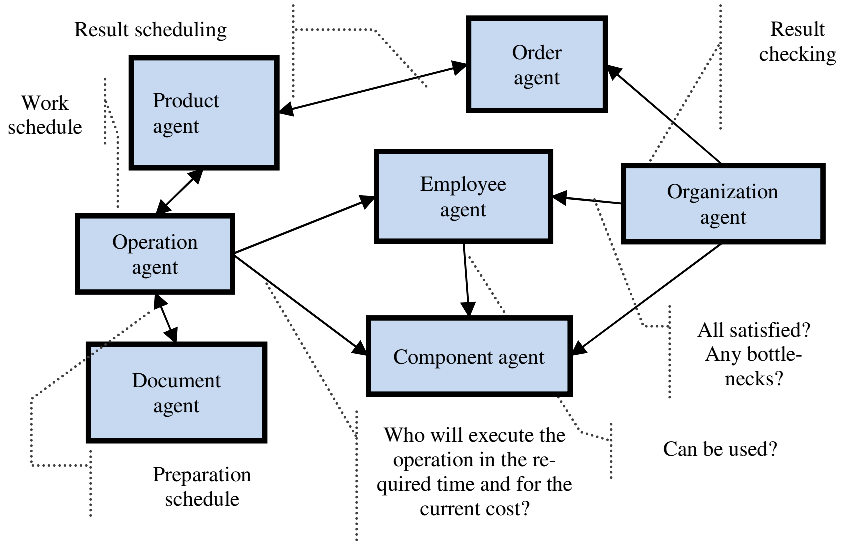


Fig. 2. Basic flows of agents' negotiation

4.3 Ontology of the Holonic Enterprise

Ontology of the enterprise allows users to specify concepts and relations of project management domain for the specific enterprises that are used by the agents considered above, for example, specific tasks are given in Tab. 2 above.

The fragment of the ontology of the enterprise considered above including the main concepts such as "project", "employee" and some other is shown in Fig. 3. Ontology allows creation of the formalized situation model for the real enterprise hereafter referred to as "scene" (Fig. 4).

The scene is formed as a network of instances of semantic concepts and relations, designed on the base of enterprise ontology connecting the classes of orders, projects, business and technological processes, operations and employees with each other. These relations are analyzed by the agents and help to work out and make decisions, restricting combinatorial search in the system.

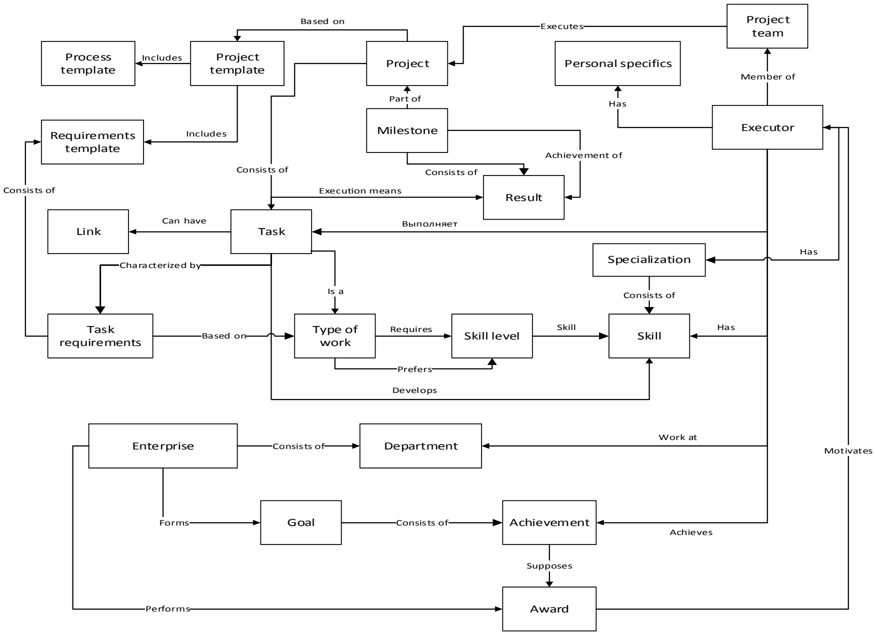


Fig. 3. Fragment of the basic ontology of the enterprise

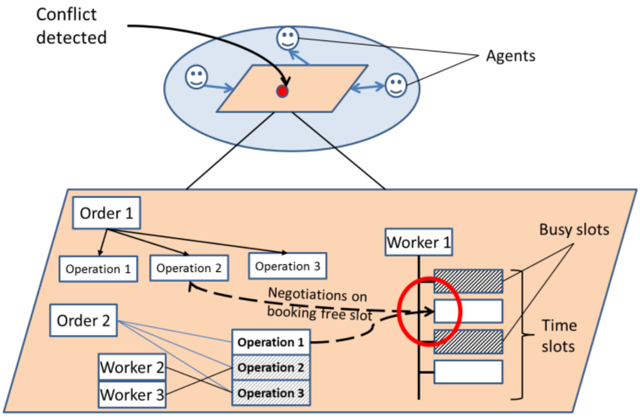


Fig. 4. Scene representation

4.4 Solution Architecture and Key Components

The solution is designed with three-tier architecture “user interface – scheduling business logic – database”, each tier can be located at a separate server.

The key component of the solution is the application server that executes the adaptive scheduling based on events, interacts with subsystems, performs data processing and provides access rights mechanisms in the system.

The subsystem of adaptive scheduling works constantly at the server side of the system and can use any relational database to store the scene. The main scheduling components are: dispatcher agent, message interaction service, agents lifecycle support service, services for creation and deletion of the agents, support of the agent communication protocol and access to scene.

5 User Interface and Workflow of Smart Enterprise Solution

5.1 User Interface

The user interface allows user to see the most important up-to-date information and easily follow changes that happen using KPI values that change in real time. The home screen provides the access to all active concepts in the system:

- projects – project details and schedule;
- tasks – current tasks execution status and results;
- employees — employees profiles and workload, virtual project shares;
- dynamics — workload reports and dashboards;
- knowledge base — ontology and scene representing the situation of the enterprise;
- services — additional services (MS Project integration etc.).

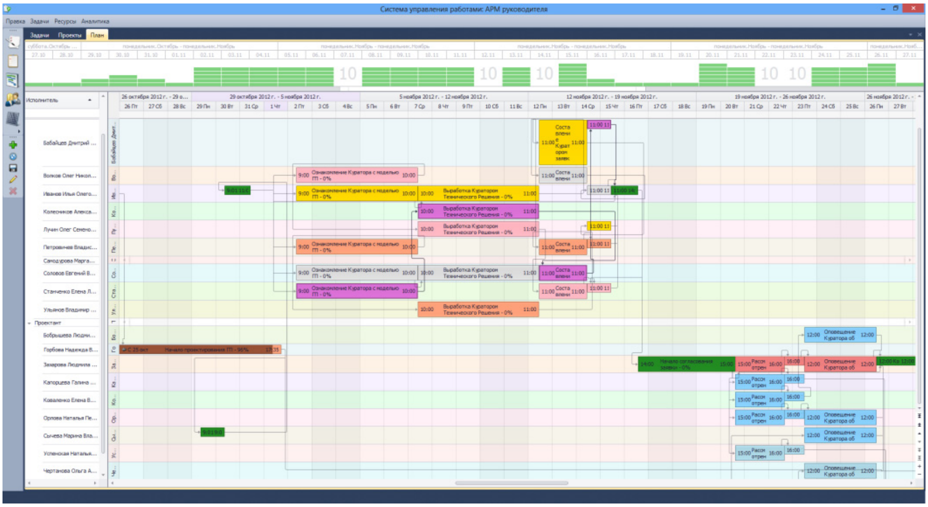


Fig. 5. The schedule represented by Gantt chart

The top-down project management is done using the Manager Workstation via the corresponding logs. The logs display the information using grid views with additional color indication that reflects task statuses (not allocated, is scheduled, in progress, delayed etc.). The visual data grid component provides functionality for grouping, filtering and sorting data by any number of fields simultaneously. The graphical representation of the project schedule over the whole resource set is done using the combined Gantt-Perth chart (Fig. 5), reflecting the dependencies between the tasks. Besides this, the top chart on the screen displays the total department workload resources lack or excess.

The most interesting component is the Developer Workstation, a new component supporting info-communication functionality of the system, which includes the following functions:

- browsing the suggested tasks and selecting the task to execute;
- task decomposition or creation of a new task, changing task parameters (for example, work estimation);
- accepting the task, starting the execution, specifying execution problems, task completion, requesting help for the task;
- ability to specify the attitude of the employee to the task, how much does he or she like it (the task, which employee doesn't like, can be reallocated);
- setting the preferences and constraints of the employee;
- event input (work left estimation, etc.);
- displaying tasks by statuses (not started, in progress, completed, etc.);
- task filtering using the semantic descriptors etc.

As a result, the system not only schedules, but also constantly communicates with the users from each project team. The goal of the communication is specifying the content of the task and possibilities to achieve the result as effectively as possible. The tasks are not "fixed" on the user, but suggested, and can be accepted or declined. The work estimation is always requested and checked using the statistics of the similar tasks executed before. The user can suggest new tasks, re-specify parameters, etc.

5.2 System Workflow

Let us consider the process of scheduling and coordination of the schedules between the developers at the level of specific tasks:

- The manager creates a new task or coordinates the necessity of the urgent execution of a task created by one of the developers;
- Parameters of the task and duration of its execution are specified according to the ontology and statistical data, but can be changed by the manager manually;
- The agent of the task finds the proper developer using its ontological description and competencies profile;
- The task comes to the Developer Workstation and asks him to consider the possibility to execute it;
- The developer selects one of the tasks form the list and sends a confirmation that he is ready to execute it. He or she can set the preferences: agree on the suggested execution schedule, specify the convenient execution date and time or even decline the task specifying the reason in order to provide information for the manager who sees all notes and can correct them and take further actions;
- If the user introduces the changes that seriously differ from the initial schedule, the agent of the task tries to find out the reasons and tries to find proper developer;

- Then the process is repeated the same way until full coordination of schedules, not allocated tasks can be compulsory assigned to the developer;
- Based on the results of the job completion, the calculation of employees wages is done according to the agreed rate, VPS are issued if the quality of the task completion and its results are confirmed by the next developer in the chain of tasks and by the project manager;
- The developer can see, how much he has earned during the day or other period and can select the tasks from the pool of unallocated but recommended project tasks to receive additional fee.

Thus, in the proposed system the tasks are scheduled, but not assigned automatically to the employees without the communication and dialog with them. During the dialog the task description can be specified, execution time can be changed, the task can be decomposed, etc. If the employee overestimates the work, the system can check it with the help of the manager or find replacement and, as a result, the developers can be stimulated to get better results.

6 Obtained Results and Prospects of the System Development

The first experience of system implementation and application for Rocket and Space Corporation “Energia” (Russia) and “Smart Solutions” software engineering company shows a high potential of the approach for increasing the efficiency of enterprise operations and support of solution by both management and employees.

The following results are expected as the outcome of the first application tests:

- increase of the employees work efficiency - by 10-15%;
- reduction of the efforts on tasks allocation, scheduling, coordination and monitoring for running project - by 3-4 times;
- increase of the reusability of the existing resources (documents, components, etc.) - from 50% and more;
- reduction of the time of response to unexpected events - by 2-3 times;
- increase of the percentage of the enterprise projects completed within the required budget and timeframe - by 15-30%;
- the platform for increasing the number of projects without increasing the number of developers and analysts.

The important result is the consolidation of team spirit within the enterprise by recognizing the message of great value of results and talents of each employee, support of individual profiles and development paths, intelligent and flexible task selection and assignment, motivation to result achievements.

The main prospects of the further system development are associated with the implementation of the network-centric p2p platform for coordination of the work of departments, the development of the adaptive scheduling method of multi-criteria decision making with self-regulation by criteria, with the use of the cloud computing to provide the developed system by SaaS model.

7 Conclusions

A new concept of holonic enterprise management is at the moment being tested at “Energia” aerospace corporation and at our own company “Smart Solutions” for increasing innovations and efficiency of resources.

The practical application of the developed concept results in the change of the company structure and the shift of paradigm from the centralized decision making and hierarchical structures to the networking organizations made on the base of business centers competing for the resources and knowledge centers competing for the projects. The important role in this structure is played by leaders (as opposed to managers) that can take the responsibility, dynamically form the teams to solve the emerging problems and achieve the results.

The first results of the software solution application show that the suggested concept becomes useful motivation tool for innovations and increase of business efficiency.

Authors acknowledge financial support of Rocket and Space Corporation “Energia” (Russia), which supports our developments.

References

1. Johnson, R.A., Kast, F.E., Rosenzweig, J.E.: The Theory and Management of Systems, 2nd edn. McGraw-Hill Book Company, New York (1967)
2. Shepard, H.A.: Changing Relationships in Organizations. In: March, J.G. (ed.) Handbook of Organizations. Rand McNally and Company, Chicago (1965)
3. Vittikh, V.A.: Introduction to the theory of intersubjective management. Samara: SSC RAS, 64 p. (2013)
4. Open Innovations. Proceedings of Moscow Forum of Innovations, Moscow (2012)
5. Koestler, A.: The Ghost in the Machine. Arcana books, London (1989)
6. Vittikh, V.A., Skobelev, P.O.: Multi-agent models of interaction to design the resource and demand networks. Automation and Telemetry (1), 162–169 (2003)
7. Vittikh, V.A., Skobelev, P.O.: Method of contiguous interactions for real time resource allocation management. Autometry (2), 78–87 (2009)
8. Skobelev, P.: Multi-Agent Systems for Real Time Resource Allocation, Scheduling, Optimization and Controlling: Industrial Applications. In: Mařík, V., Vrba, P., Leitão, P. (eds.) HoloMAS 2011. LNCS, vol. 6867, pp. 1–14. Springer, Heidelberg (2011)
9. Leung, J.Y.-T. (ed.): Handbook of Scheduling: Algorithms, Models and Performance Analysis. Chapman & Hall/CRC Computer and Information Science Series (2004)
10. Voß, S.: Meta-heuristics: The state of the art. In: Nareyek, A. (ed.) ECAI-WS 2000. LNCS (LNAI), vol. 2148, pp. 1–23. Springer, Heidelberg (2001)

Advanced Shop-Floor Scheduling with Genetic Algorithm for Combined Horizon Optimization in Holonic Manufacturing Systems

Cristina Morariu, Octavian Morariu, and Theodor Borangiu

University Politehnica of Bucharest, Dept. of Automation
and Applied Informatics, Bucharest, Romania, 060042
{cristina.morariu,octavian.morariu,
theodor.borangiu}@cimr.pub.ro

Abstract. Shop-floor scheduling is one of the most complex problems in holonic manufacturing systems as it deals with unpredictable scenarios and overlapping requirements. The scheduling problem is considered to be a NP-hard problem and only near optimal solutions can be obtained. A solution for this problem can be formulated only by employing meta-heuristic class algorithm. This paper discusses the scheduling problem in heterachical operating mode, focusing on solving the local horizon problem. A distributed genetic algorithm is introduced, that uses the local operation plan to generate the initial solution population. The initial population is then evolved based on global optimum soft conditions until the acceptable solution fitness is achieved. The soft conditions considered at global horizon layer are energy footprint, resource utilization and supply chain optimization for resource stocks. The paper describes the data structures used to model this logic and describes in detail the genetic algorithm evolution mechanism. Experimental results are discussed in the context of a pilot production line consisting of six universal resources and a conveyor belt processing two parallel customer orders.

Keywords: Shop-floor scheduling, genetic algorithm, heterachical operation, local horizon, intelligent product, holonic manufacturing systems.

1 Introduction

Shop floor production scheduling remains one of the most important problems in holonic manufacturing systems domain, mainly because of the intricacies of determining the optimum solution for each particular scenario that can arise during production. An intelligent production scheduler must fulfil at the same time functional requirements that depend on the shop floor configuration and operational requirements imposed by the upper layer systems integration. Functional requirements are dictated by the shop floor design, mentioning resource capabilities, product complexity and conveyor layout and others. These aspects are usually highly dependent on the actual

V. Mařík, J.L. Martinez. Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 25–36, 2013.

product design that is manufactured. This is one of the reasons why a general solution cannot be formulated for the scheduling problem in manufacturing. Operational requirements refer mainly to parallel customer orders, or parallel batch manufacturing, also referred in literature as rush orders, where the scheduler must be able to keep track of several product batches within their make-span and assure the completion is in the timeframe agreed. Another operational requirement is robustness of the system, where the scheduler must adapt to unpredictable events that might occur during the manufacturing process, like resource breakdown or stock depletion. These operational requirements can be abstracted to the concept of operational flexibility that introduces a more concrete requirement for the scheduler: the capability to dynamically reschedule the operations at runtime without external intervention. In literature this has been coined as the heterarchical operational model, where products and shop floor resources interact, usually through a mediator, in order to complete the batch production when the initial plan cannot be completed due to unexpected events. This has been discussed by Borangiu et al. [1]. To improve the flexibility of the system several concepts have been proposed like: intelligent products, multi agent systems in support for distributed decision making by Shen et al [5], service orientation at shop floor by Borangiu et al [2], standardization of communication and so on. The main goal is to distribute the intelligence across all actors involved in the production process, mainly products on the production line and resources. However, direct interaction between products and products on one hand, and products and resources on the other, introduces a concurrency problem, as the local goal of each product is to complete its remaining operations. This problem can be seen as a limited horizon problem, where the local optimum that can be computed is different from the global optimum. As a compromise solution, the concept of a mediator entity has been introduced, which gathers information of all resources and products, and so achieves a higher horizon. At this layer certain optimizations can be implemented, specifically in regards to prioritization of products in regards to resources. However, the mediator entity is agnostic over the customer orders, specialty when parallel customer orders support is considered. With this reasoning in mind, three types of optimization can be considered for systems that are capable of functioning in heterarchical mode: local horizon optimization mediated horizon optimization and combined horizon optimization. Combined horizon optimization refers to the ability of the scheduling mechanism to consider both local criterion for the solution and global constraints, like resource utilization, energy footprint and supply chain optimization by including stock availability in the scheduling process.

This paper introduces an algorithm that aims at combined horizon optimization for shop floor scheduling based on a distributed genetic algorithm. The advantages of this approach are the ability to generate a global optimum schedule in real time, based on local horizon optimization collected from intelligent products and resources and evolved using genetic algorithm approach.

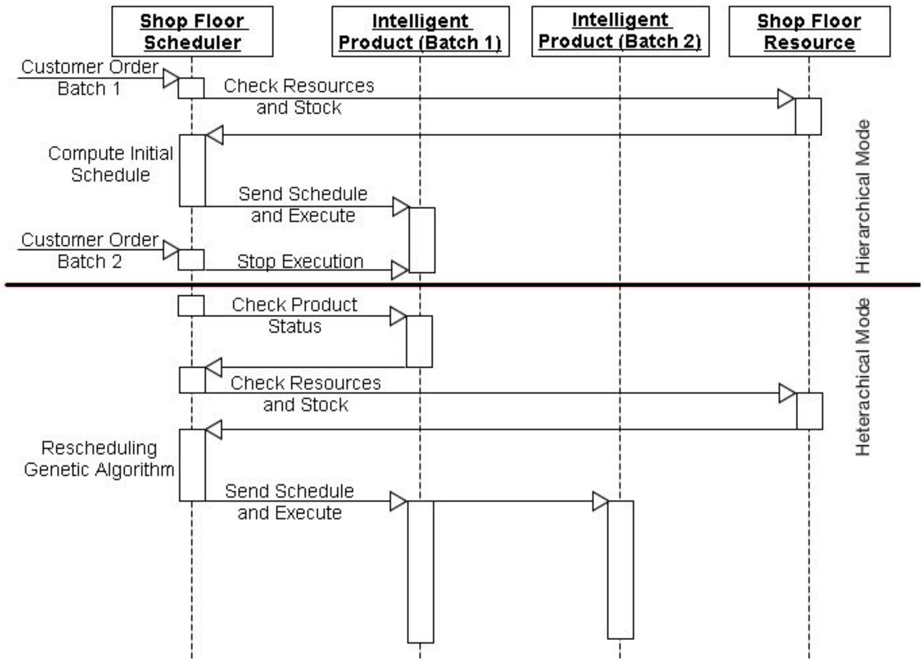


Fig. 1. Data Flow for Hierarchical and Heterachical Mode

This approach proves to be superior compared to the basic local horizon approach especially in regards to global KPIs like energy footprint, global resource utilization and supply chain integration. Overall, the approach proposed in this paper improves system predictability in heterarchical operation.

2 Related Work

Various solutions for solving the scheduling problem for job shop resources have been proposed in the literature over the last years. The solutions are generally proposing meta-heuristic class of algorithms capable of generating a near optimal solution. Dorigo et. al. [8] proposed an ant-colony algorithm for multi agent systems. The approach tries to mimic the social behavior of ants to accomplish a complex behavior of the agents on the job shop. An alternative bee colony optimization algorithm to job shop scheduling is proposed by Chong, Chin Soon, et al [10]. Nowicki et. al. [9] introduces a tabu-search algorithm for job shop scheduling problem. Genetic algorithms have been proposed for the scheduling problem at different levels before [11, 12, 13, 14] and compared with other approaches have proved to be a viable solution. The GA proposed in this paper is focused for the holonic architecture described in PROSA [15]. The GA algorithm proposed is using holonic data structures for the solution population.

3 Holonic Manufacturing System Design and Information Flow

A service oriented holonic manufacturing system generic architecture as coined by Borangiu et al [1, 2] is represented in Figure 2. In the context of holonic manufacturing systems and specifically considering SOA orientation, information flow and availability at runtime is equally important as the material flow. Figure 1 presents the information flow in the manufacturing system in hierarchical and heterachical operation modes. The process starts with the customer order (Order Holon/OH), or in other words the first product batch. The shop floor scheduler checks the resources available in the manufacturing system and computes the execution schedule for each product. The schedule, represented by the Aggregate Product Order (APO) is sent to each Product Holon (PH) and the execution in hierarchical mode starts. When a second OH arrives, considered a rush order, meaning that the execution cannot be postponed until the first OH is completed, the shop floor scheduler switches to heterachical operational mode, by stopping the execution of the products from the first batch.

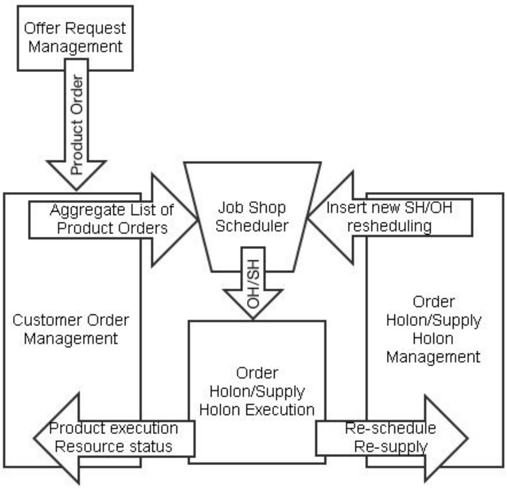


Fig. 2. Holonic Manufacturing System Architecture

The heterachical mode rescheduling starts by collecting the current execution status from all PHs on the production line. In other words, the PH will send back to the scheduler the list of completed operations and the list of the remaining operations, together with the current position of the pallet on the manufacturing line. The next step consists in checking the resource status and the raw material stocks available for each resource. Once this information is gathered by the scheduler, a new schedule is computed using a genetic algorithm that considers both the first batch and the second batch. The new schedule is sent to all the products in both batches and the execution is resumed. The system continues to function in heterachical mode until all concurrent OHs are completed. The following sections describe the data structure used by the PH, the shop-floor resources and scheduler.

3.1 Product Data Structure (Product Holon)

The intelligent product is formed by the association between the product pallet which is initially empty and the information regarding the product that will be manufactured. This is represented in the system by the Product Holon (PH). The intelligent product concept has been studied by McFarlane et al [4] and Wong et al [7]. Mayer et al [5] published a comprehensive survey on intelligent products in industry. The product pallet is equipped with an embedded device capable of WI-FI based communication, web services support for accessing data and a read/write persistent storage for data. The informational part of the PH consists in details about the product identity, like RFID tag, product type and most importantly the product recipe. The product recipe is the list of operations and precedencies between them, required to manufacture the product. In hierarchical operation mode this list contains also the shop-floor resources where the operations will be executed, while in heterarchical mode these resources will be computed in real time. The data structure for the information stored in the PH embedded memory is presented in Figure 3. The operation dependency is always noted backwards, or in other words operation X+1 may depend on operation X, X-1 and so on.

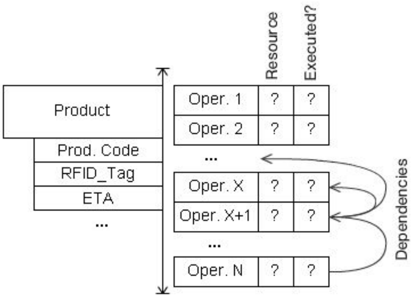


Fig. 3. Product Data Structure (PH)

The PH offers Web Service based APIs that allow uploading, updating and downloading of the data structure at runtime. These APIs are accessed by the scheduler for upload and download, and by the shop floor resources for update after each operation that is completed. The information available for each intelligent product represents the local horizon of the product. In other words, the horizon is limited to the next operation that needs to be executed, considering the operations already executed and the dependencies of the next operation.

3.2 Resource Data Structure (Resource Holon)

The Resource Holon (RH) structure consists in the list of operations that the resource can execute. These operations have associated a stock representing the raw materials required, and the energy footprint required to perform the operation, as illustrated in Figure 4.

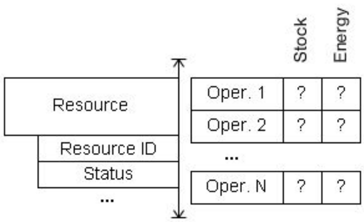


Fig. 4. Shop floor resource Data Structure (RH)

The RH provides Web Services APIs that allow checking the stock and availability. These APIs are called by the shop floor scheduler during the initial scheduling and for each rescheduling operation in heterachical mode. Internally the resource updates the stock information after each operation. The energy footprint for each operation can be a pre-set constant, or can be updated by the resource dynamically after each execution of that operation. This depends largely on the self-monitoring capabilities of the resource. The resource, or the resource mediator which consolidates the data from all resources, has a larger horizon than the intelligent product, as it could consider parallel requirements from products on the production line competing for the same resource. This is considered the mediator horizon.

3.3 Schedule Data Structure (APO)

The shop-floor scheduler uses a set of data structures to represent the problem solution (APO), each representing a product batch, as illustrated in Figure 5. The scheduler horizon spans over all the OHs in execution at a given time in hieratical mode. When switching in heterachical mode, the scheduler collects the current information available in the PHs, reflecting the local horizon and in the RHs, reflecting the mediated horizon, and assembles it in the data structure presented in this section. This allows the creation of the combined horizon, in which both the current state of the product and the shop floor resources are reflected. At this point, the data structure will be completed only with the operations that were already executed for each product in the batch, the location of the products on the conveyor and the status of the shop floor resources. This represents a partial solution. Starting with these facts, the scheduler will merge the new customer order with the current data and will generate an execution solution that will assure the execution for the remaining operations for each product already in the production line, and scheduling of the products for the new OH. The solution generated is similar in terms of the structure with the initial APO, the main difference being the fact that the un-executed operations are rescheduled. The scheduling problem is a NP-hard [8] problem and so only near optimal solutions are possible. To resolve this problem, a meta-heuristic algorithm in the genetic algorithm class is used. The algorithm is discussed in the next section.

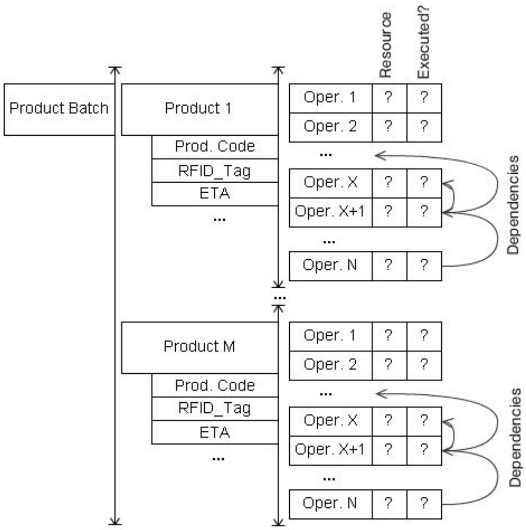


Fig. 5. Scheduler Data Structure (APO)

4 Genetic Algorithm Design

There are several hard conditions that need to be respected in order for a computed schedule to be accepted as a viable solution such as operation precedence, raw materials stock availability, resource availability for all required operations and completion time within the required due date. As these conditions are sometimes conflicting a genetic algorithm would be able to generate only sub-optimal solutions for the scheduling problem. Genetic algorithms start with a randomly generated population of solutions and by applying operations as selection, crossover and mutation on individuals, create new generations evaluating the fitness of each individual of the population in the process. When the fitness level in the population reaches a satisfying value, a set of solutions are obtained.

4.1 Initial Population

The initial solution population is generated based on the information collected from the intelligent products that are in work at the time of re-scheduling as illustrated in Figure 6. The green operations are the operations that were executed for each product so far. The red operations are the operations that still need to be executed in order to complete the product. The initial population is generated by randomly allocate resources to the remaining operations for each product in the first batch. For the second batch, which does not have yet any operation executed, the operation allocation schedule will be completely random. Each individual in the population generated will have two or more instances of product batch data structure, with at least one partially completed. The information gathered from the products, illustrated

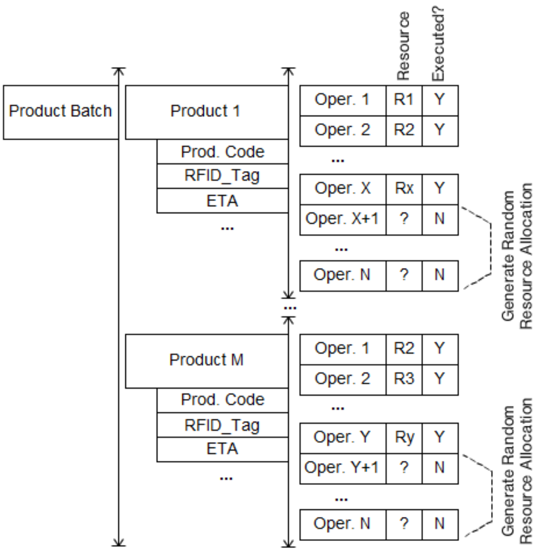


Fig. 6. Initial Population Generation

in green in Figure 5, represents the local horizon, or in other words, the next operation horizon and is included in the algorithm from the initial population generation.

4.2 Fitness Function

The fitness function is used to evaluate each individual in the solution population and determine a score. The fitness is computed by evaluating a set of conditions against the individual scheduling solution instance as follows:

Condition 1 (Hard Condition) – Resource Availability: checks if the resources are available for the operations that were randomly allocated. In order for this condition to be fulfilled the resource must be able to do the operation and the local raw material stock on the resource must be available. If this condition is fulfilled 50 points are awarded to the individual.

Condition 2 (Hard Condition) – Operation Precedence: checks if the order of the operations in the individual solution respects the dependencies of each operation for each product. If all precedencies are fulfilled 50 points are awarded to the individual.

Condition 3 (Hard Condition) – Time Constraint: refers to the completion time for each product batch, corresponding to a customer order. The individual solution must assure completion within the expected make-span for each product batch. If this condition is fulfilled 50 points are awarded to the individual.

Condition 4 (Soft Condition) – Conveyor Travel Distance: refers to the total distance on the conveyor where the products are transported. The goal is to keep the total transportation distance of the products to a minimum. The distance is computed from the sequence of resources allocated for operations for each product and then summed up at batch level. Depending on the value computed in regards to the maximum and minimum value from the whole population, a range between 0 and 10 points are awarded.

Condition 5 (Soft Condition) – Energy Footprint: refers to the total energy footprint for the product batch manufacturing. This is summed up from all operations scheduled to be executed on resources, at product level and then at batch level. Depending on the value computed in regards to the maximum and minimum value from the whole population, a range between 0 and 10 points are awarded.

Condition 6 (Soft Condition) – Resource Utilization: refers to the overall resource utilization. The goal is to have high and distributed resource utilization. The resource utilization average is computed by counting the operations scheduled for each resource. Two thresholds are used to limit the resourced utilization to the desired level. If the resource utilization computed is between the two thresholds 5 points are awarded to the individual.

Condition 7 (Soft Condition) – Stock Optimization: refers to the stocks available on each resource. The goal is to assure stocks are equally consumed from each resource in order to add predictability in the supply chain process. The resource stock differences for each operation are evaluated for each resource that can execute that operation. If these differences are less than 20%, 5 points are awarded to the individual solution.

The total score computed based on above evaluation is divided to the maximum possible score to obtain a fitness value between 0 and 1, for each individual.

4.3 Selection, Crossover and Mutation

Genetic algorithms have typically three operations: selection, crossover and mutation [15]. The selection operation consists in computing the fitness value for each individual in the population and sorting the population based on the results. Then, the best 65% individuals are selected for crossover operation. The crossover operation represents the combination of two Schedule instances that produce an offspring. The crossover operation is implemented by generating a random number X (crossover point) between 1 and N, where N is the number of operations. The offspring will inherit the schedule of the first parent from operation 1 to operation X and the schedule of the second parent from operation X+1 to operation N. Figure 7 represents this crossover operation. The crossover operation is applied to each product individually in the solution. This is because the crossover point range is different for each product, depending on how many operations are already completed.

The mutation operation is applied to a randomly chosen subset of individuals in each generation, and consists in rescheduling of one operation from the data structure. The new schedule is generated by randomly assigning a new resource for the selected operation instance. Both the operation and the new resource are selected and generated randomly.

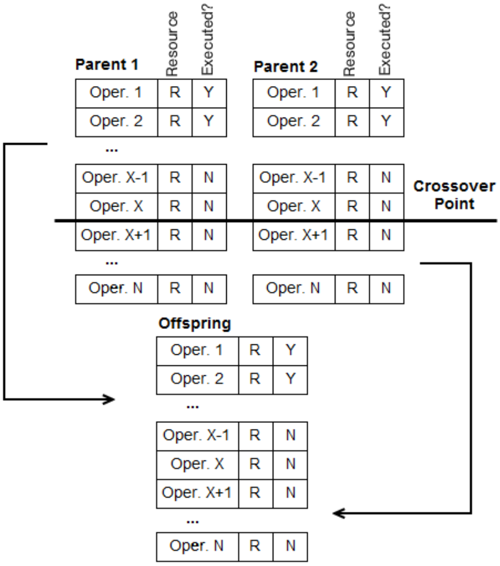


Fig. 7. Crossover Operation

4.4 Algorithm Steps

The genetic algorithm has the following structure:

```
Step1: generateInitialPopulation()
Step2: while (best individual fitness < min_fitness){
Step3: do_crossover(best 65% individuals)
Step4: calculate_fitness(offsprings)
Step5: remove_worst(worst 35% individuals)
Step6: calculate_best_individual_fitness
Step7: }
```

As we can see in the above pseudo code, the genetic algorithm sorts the individuals after the crossover operation, based on the fitness and removes the worst individuals from the population. This assures the evolution from one generation to another.

5 Experimental Results

To validate the scheduling algorithm proposed in this paper the following scenario has been simulated: a manufacturing system composed of six universal resources connected by a conveyor belt capable of transporting product pallets circularly. Each resource has a private workspace where product pallets can be elevated from the main conveyor belt. The scenario involves assembly of T shaped products from raw materials, with a first customer order of 20 products followed by a second customer order (rush order) of 10 products. For each product assembly six operations are required.

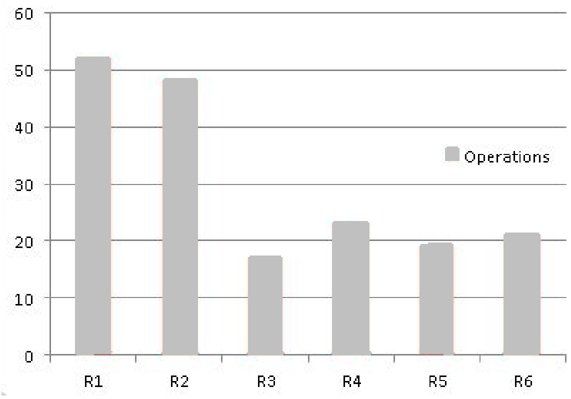


Fig. 8. Operation Distribution on Resources

Figure 8 illustrates the overall allocations of operations for each resource in the manufacturing system for a total of 180 operations, 120 operations for the first batch and 60 operations for the second batch. We can see that the algorithm tends to assign more operations to the first two resources and that the distribution on the remaining four is relatively equal. This is explained by the fact that R1 and R2 are configured in this simulation with a lower energy footprint associated with the operations and so condition 5 takes precedence over soft condition 6 as the score is higher. However, for R3, R4, R5 and R6, that have the same energy footprint, the distribution of operation schedule is similar. The algorithm was executed in this scenario with a population size of 100 individuals and the evolution was considered with a maximum of 10000 generations, considering the problem size of 30 products with 6 operations each and 6 resources available. The min_fitness threshold was configured at 0.95. The execution time for this scenario was under 50 seconds for 100 different executions monitored.

6 Conclusions

Shop floor scheduling remains a complex problem due to the strong dependency between the shop floor design and the algorithms used. Genetic algorithms prove to be an efficient approach for designing a shop floor scheduler as the conditions can be easily adapted to fit a specific scenario. Another advantage of using genetic algorithms for shop floor scheduling is the unique ability of these classes of algorithms to incorporate the combined horizon problem that generally affects the local decision approach even in mediated systems. The experimental results show that the algorithm is able to generate schedules that not only are valid but also are energy efficient and assure uniform resource utilization Future work is focused on the study of the algorithm behaviour in more complex simulation environments, specifically in situations where the product batches contain mixed product types. Preliminary simulations show a high increase in execution time with a larger problem size: 100 products with 10 operations on 10 resources with a population size of 100 over 10000 generations and 0.95 min_fitness setting take in the range of 8 minutes to execute and generate a solution on similar hardware.

Acknoledgements. This work was partially supported by the European project Empowering Romanian Research on Intelligent Information Technologies (ERRIC), FP7-REGPOT-2010-1/264207.

References

1. Borangiu, T., Gilbert, P., Ivanescu, N., Rosu, A.: An Implementing framework for holonic manufacturing control with multiple robot-vision stations. Journal of Engineering Applications of Artificial Intelligence (2009) ISSN 0952-1976
2. Borangiu, T., Raileanu, S., Anton, F., Parlea, M., Tahon, C., Berger, T., Trentesaux, D.: Product-driven automation in a service oriented manufacturing cell. In: Proceedings of the Int. Conf. on Industrial Engineering and Systems Management, IESM 2011, Metz, May 25-27 (2011) ISBN 978-2-9600532-3-4
3. McFarlane, D., Sarma, S., Chirn, J.L., Wong, C.Y., Ashton, K.: The intelligent product in manufacturing control and management. In: 15th Triennial World Congress, Barcelona, Spain (July 2002)
4. Meyer, G.G., Främling, K., Holmström, J.: Intelligent products: A survey. Computers in Industry 60(3), 137–148 (2009)
5. Shen, W., Wang, L., Hao, Q.: Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 36(4), 563–577 (2006)
6. Wong, C.Y., McFarlane, D., Ahmad Zaharudin, A., Agarwal, V.: The intelligent product driven supply chain. In: 2002 IEEE International Conference on Systems, Man and Cybernetics, vol. 4, p. 6. IEEE (October 2002)
7. Blazewicz, J., Domschke, W., Pesch, E.: The job shop scheduling problem: Conventional and new solution techniques. European Journal of Operational Research 93, 1–30 (1996)
8. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 26(1), 29–41 (1996)
9. Nowicki, E., Smutnicki, C.: A fast taboo search algorithm for the job shop problem. Management Science 42(6), 797–813 (1996)
10. Chong, C.S., et al.: A bee colony optimization algorithm to job shop scheduling. In: Proceedings of the Winter Simulation Conference, WSC 2006. IEEE (2006)
11. Biegel, J., Davern, J.: Genetic Algorithms and Job Shop Scheduling. Computers and Industrial Engineering 19(1), 81–91 (1990)
12. Davis, L.: Job Shop Scheduling with Genetic Algorithms. In: International Conference on Genetic Algorithms and Their Applications, Pittsburgh, PA, pp. 136–140 (1985)
13. Falkenauer, E., Bouffouix, S.: A Genetic Algorithm for Job Shop. In: IEEE International Conferernce on Robotics and Automation, Sacramento, CA, April 9-11, pp. 824–829 (1991)
14. Kanet, J., Sridharan, V.: PROGENITOR: A Genetic Algorithm for Production Scheduling. Wirtschaftsinformatik 33(4), 332–336 (1991)
15. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference architecture for holonic manufacturing systems: PROSA. Computers in Industry 37(3), 255–274 (1998)

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

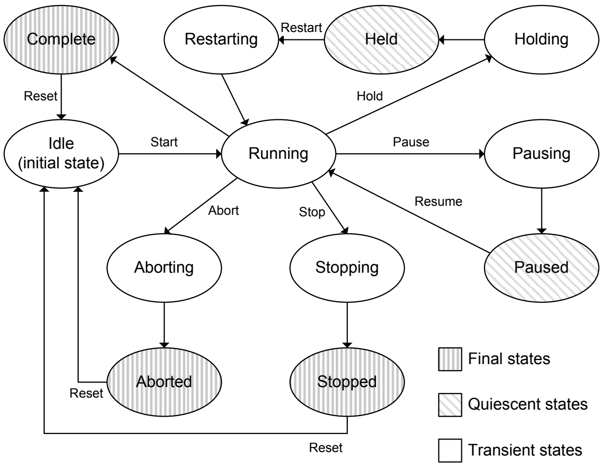


Fig. 1. State machine model of the standard IEC 61512 [13]

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

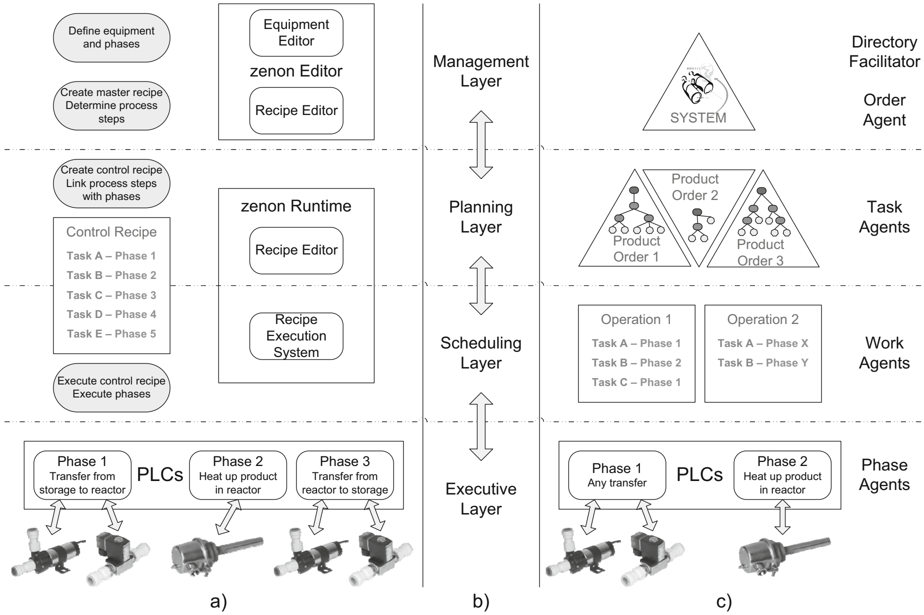


Fig. 2. Structure of an industrial batch management system (a), specified layers (b), and architecture of the MAS (c)

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

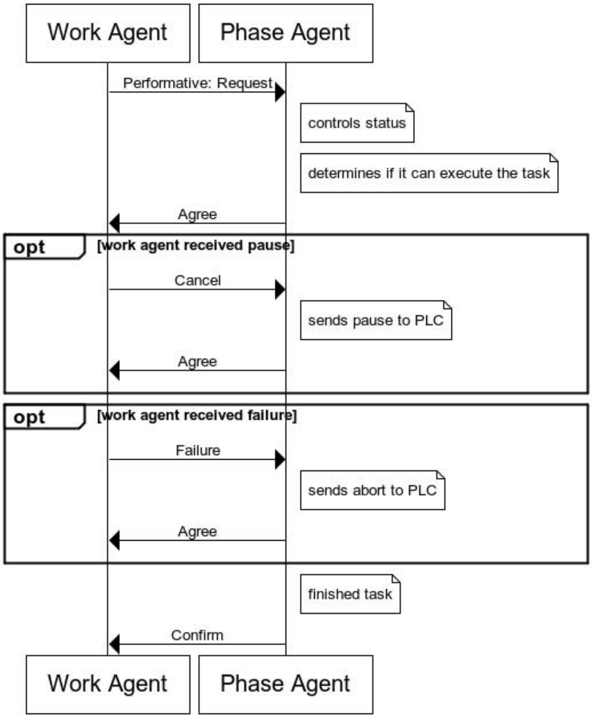


Fig. 3. Example communication between work agent and phase agent

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.

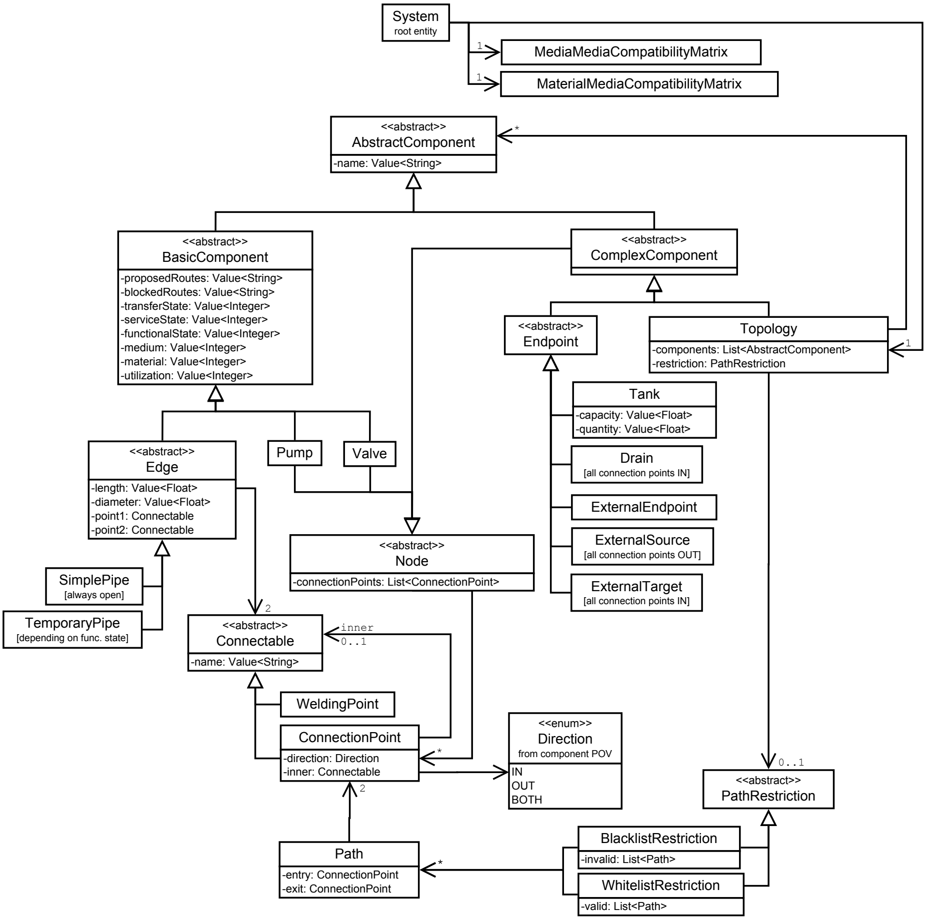


Fig. 4. Class diagram of the ontology defining the representation of the physical system components

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.

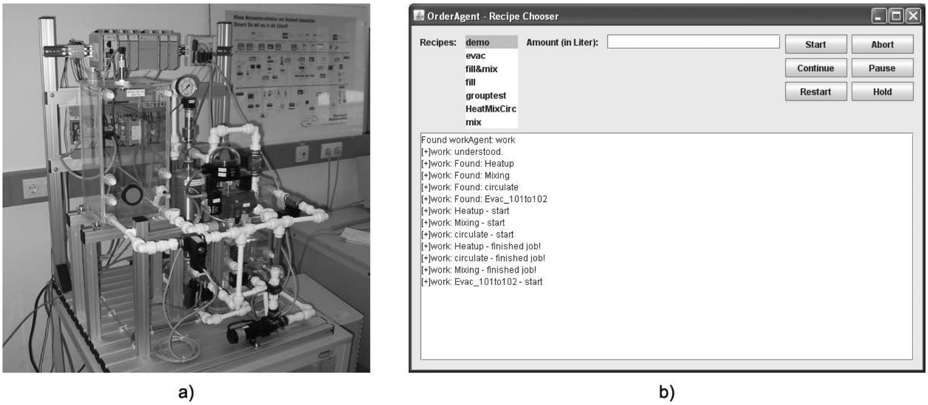


Fig. 5. The laboratory process plant (a) and a screenshot of the graphical user interface provided by the order agent (b)

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

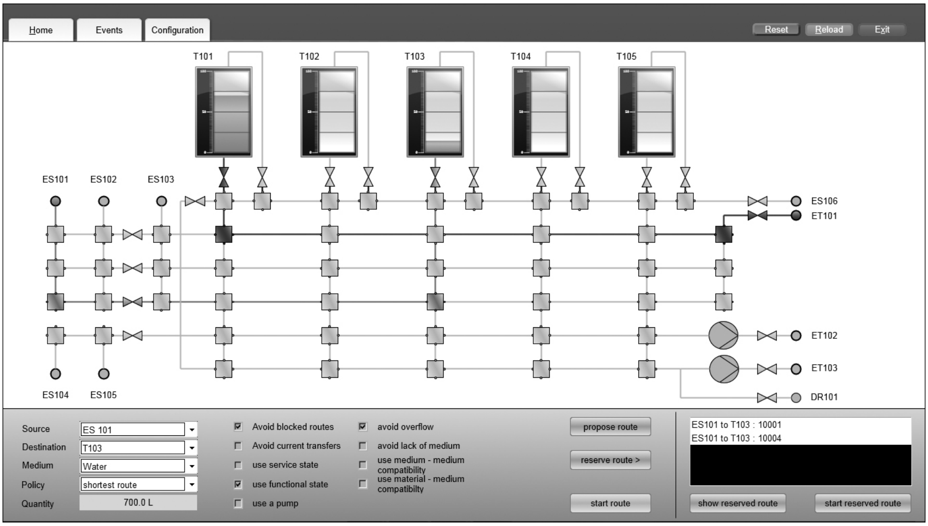


Fig. 6. Visualization in zenon with transfer tasks in execution

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)

Production Plan-Driven Flexible Assembly Automation Architecture

Alois Zoitl, Gerd Kainz, and Nadine Keddis

fortiss, An-Insitut der Technischen Universität München,
Guerickestr. 25, 80805 Munich, Germany
{zoitl,kainz,keddis}@fortiss.org
http://www.fortiss.org

Abstract. Manufacturing industries are currently under a strong pressure to easily adapt to changing market situations. In order to stay profitable production automation systems need to flexibly adapt to different products, product variants, and product volumes. In this work we investigate how a flexible recipe-based control approach can be transferred from batch automation systems to discrete manufacturing. We define a generic control architecture and a manufacturing recipe model that allows to execute recipe parts directly in the low-level control devices of the involved manufacturing cells. An evaluation of the developed system on a demonstration plant shows the aptness for discrete manufacturing. The developed system with its flexibility on the lowest control level can also serve as a foundation for highly flexible supervising control strategies like the HMS approach.

Keywords: Recipe-based control, discrete manufacturing, flexible adaptive production.

1 Introduction

Nowadays, production industry is going through a transformation. The demand for adaptable manufacturing systems has increased due to changes in the market. There is a shift away from mass production towards mass customization. This is coupled with dynamic requirements regarding lot size, product variants, lead times, and costs which emphasize the need for an easy adaptation of the production system. Moreover, product life cycles have become shorter, leading to the need for a fast changeover process. Additionally, product quantities vary largely. Smaller lot sizes for each product variant at an increased number of variants per product is becoming the standard for the production industry. Therefore, key features of future manufacturing systems should include adaptability while minimizing setup times and increasing the transformability of the factory [8]. At the same time, it should be possible to produce different variants of different products simultaneously.

Current manufacturing systems contain a number of sensors and actuators that are part of a digital network. The processes in a factory involve several

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 49–58, 2013.

machines inside a factory as well as on different sites. Changes in the system require a lot of mechanical work [18] and the reconfiguration of information technology (IT) systems. Current approaches to overcome these problems mainly focus on mechatronic compatibility to minimize configuration effort and time needed for applying any changes to the factory. However, IT systems need to be reconfigured as well to integrate new components and adapt to the applied changes. To achieve flexible and adaptable production, IT systems have to be modularized as well [11].

Looking at how the production is planned today, the observation is that production plans are fixed and optimized for the efficient production of a single product. Production plans are rarely changed during production. The changeover process for producing new products requires a stop of the production before some of its parts have to be reconfigured, reprogrammed, or even replaced. This is not always possible in current systems. However, future manufacturing systems need to be reconfigured quickly to keep up with the fast pace of changes in markets.

IT is becoming more and more important for manufacturing [17], and so the widely used concepts from the IT domain are introduced to the manufacturing domain. The concept of cyber-physical systems is an example for this which brings many potentials and benefits such as interoperability and modularity as well as improvements on the engineering level [1]. Using this concept, modules that are part of the manufacturing system are getting more intelligent and interact with other modules to form a larger system. For the manufacturing industry this means that subsystems have the ability to act (partially) autonomously to interact with other subsystems to produce different goods.

Configuring the IT system requires information about the setup of the factory, the logistics, the production orders, and the production plans as illustrated in Figure 1. Currently, all of this information has to be added manually to the IT system in order to reflect any change in the factory. Expert knowledge is necessary to add the information at the right place in the IT system as well as to understand how this information is propagated within the system. This is very costly and time consuming. Additionally, the software of the different machines has to be programmed to match the production plans and orders. Information about changes in the factory as well as information about the logistics can be automatically integrated using self-description mechanisms [7]. However, machine programs still have to be manually written and the production plans adapted.

Therefore, we propose a production plan-driven architecture to automatically generates the needed machine programs from the production plans and orders using the information about the factory setup. The goal of this approach is to minimize the programming effort related to changing production plans to produce different products or variants. For this, the software of the subsystems, in our case the different stations, has to be modularized. Each module that can be used in the system provides its software with defined interfaces that can be used by the control program. Ideally, these interfaces are standardized so that they can easily be replaced by other modules and can be accessed using the same mechanisms throughout the whole program. Each module then provides a

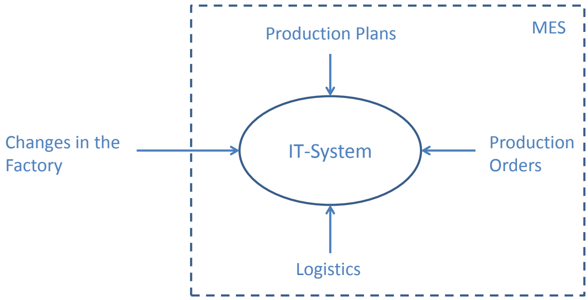


Fig. 1. The required information to configure the IT system of a manufacturing system

set of commands that can be used to invoke different production steps. These production steps match the ones that are typically used in production plans to enable an automatic execution of the production plan for the desired good.

The paper is structured as follows: Section 2 provides an overview of existing approaches and related work. We will then explain the suggested control architecture in Section 3 followed by the definition and execution approach for recipes in Section 4. We evaluate the approach on an industrial production system used for educational purposes that is described in Section 5. Section 6 summarizes the paper and gives an outlook on future work.

2 Related Work

In the last years much work has been spent on developing design patterns and improving control software development. The focus hereby has been on increasing the software quality, reusability of control software modules, and adaptability (see for example [6, 13–15, 20]). However, these approaches mainly target rigid machine or program structures. They consider variability only between different machines. In contrast, we target handling different products and product variants on the same machine and adapt the machine's operation to it. Nonetheless, these works on modularizing control application can serve as a basis for this work as they help to identify independent units of operation within machines.

In order to increase the flexibility of batch process automation plants the International Standardization Association (ISA) developed the standard ISA 88 [3]. The key element in this standard is the separation of production equipment control from the recipes of the produced product. Especially for the equipment standardized operational state machines have been defined. These can than be used by a central batch management systems for coordinating the equipment according to the product recipes. The Open Modular Architecture Controls (OMAC) association has taken this concept also to discrete manufacturing. In their PackML guideline [2] they transferred the ISA 88 equipment state machines to the requirements of packaging machines. The new product for component oriented development of automation applications from the company 3S called

Application Composer[1] also utilizes this state machine for structuring machine control sequences. While this defined machine operation is a great benefit for coordinating a line of machines it leaves open how to handle product specific production data.

While ISA 88 mainly targets the flexible application of different recipes (i.e., different products or product variants) on a more or less fixed plant structure the Holonic Manufacturing Systems (HMS) approach intends to break up the rigid structures and further increase flexibility and adaptability [4]. However, similar to the ISA 88 approach also in HMS manufacturing components equipped with basis automation providing the basic functionality of the manufacturing components are coordinated [5]. In HMS product plans are mainly used for the coordination and sequencing of production equipment [9, 12, 19]. How to structure the basis automation of the manufacturing components and how to use product plan data has so far not been considered.

3 General Architecture

The base for our recipe-based control system is a modularization of the underlying PLC program. Therefore, a module has been created for each type of component controlled by the PLCs. A module comprises a set of functions used to control the component. Since a module is available for each component type, the control of the system is raised to a higher level of abstraction. Instead of controlling the system on the level of input and output signals, the application developer can use modules to program on a function level (e.g., move cylinder to extended position, turn on conveyor belt with direction forward). This already improves the speed of application development as the function abstracts a lot of details away from developers.

To reuse modules even if multiple components of the same type are controlled by one PLC, the access to the I/O pins used to control a specific component have been refactored into a module instance configuration. This configuration is handed to all module functions upon invocation allowing the module functions to control a specific module instance. This makes the modules implementation independent of a concrete module instance and reduces the effort of providing the implementation for a module instance to the creation of its configuration. The configurations are data structures providing a set of fields describing the I/O pin configuration of the module instance. Depending on the number of I/O signals used to control a module instance, a module configuration can range from a few to many fields. For example, the configuration of a conveyor belt might consist of an enable and a direction field with the values Q0.0 and Q1.2. The introduction of module configurations helps in creating independent module implementations which are parameterized with the configuration of a module instance upon execution.

[1] Application Composer: `http://www.codesys.com/products/codesys-engineering/application-composer.html`

The next step to improve the reusability of the system is the introduction of interfaces. The concept of interfaces has been borrowed from object-oriented programming languages and enables grouping a set of functions. If a module now supports an interface it is required to provide implementations for all the functions defined by the interface. So the usage of interfaces allows the programming of control programs based on the contracts defined by the interfaces. This enables system developers to start with the implementation of the overall system before the implementation of all component modules is finished. As long as the system developers only call functions defined by the interfaces implemented by a component everything will work upon integration. On the other side, the component developers only need to provide the implementations for the interface functions to ensure a seamless integration in the overall system without having detailed knowledge about the remaining system.

Interfaces allow to abstract from a specific component implementation. For example, if two different kinds of motors (e.g., a stepper motor and a servo motor) implement the same interface they can be interchanged without changing the control program. In addition, if someone knows how to control a motor of type A via the motor interface he can immediately control a motor of type B if it provides an implementation for the same interface.

Modules and interfaces are the foundation of our recipe-based control as they provide a first abstraction layer from the underlying system. Together they allow to program a system based on generic control primitives instead of always dealing with the concrete implementation of the components. On top of the modules with their interfaces is the recipe controller. The recipe controller is a component responsible for the realization of the control program driving the module instances. Therefore, it takes a recipe as input and executes it. By loading different recipes into the PLC the control program can be easily exchanged without having to reprogram the whole PLC. This gives a high level of flexibility required for the realization of adaptable manufacturing systems.

To complete the architecture and ensure a correct execution a startup code is required. This startup code is called program outline. It takes care of the basic control of a station including its locking mechanisms and triggers the recipe controller for the execution of the currently loaded recipe. To reuse the program outline among the implementation of different stations, hooks are used. Hooks are places in the program where the execution of station related code is required which is not known by the generic program outline. For the hook realization we rely on the call of predefined functions which have to be implemented with respect to the details of a concrete station.

Figure 2 shows the a schema of the system architecture consisting of the program outline, recipe controller, and a storage for the recipes. As can be seen in the figure the control center directly interacts with the program outline for basic control (e.g., start and stop) and the recipe controller to load the recipe for the production.

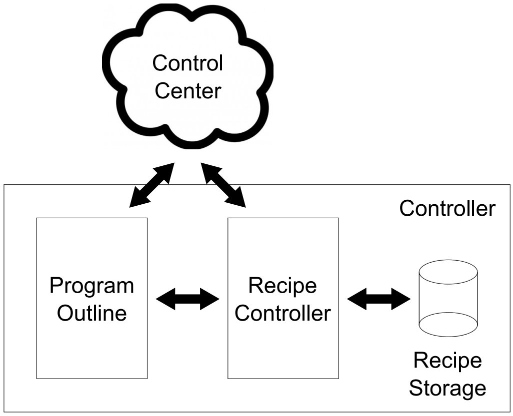


Fig. 2. Architecture of the Execution System Based on Recipes

4 Recipe Definition and Execution

Based on the generic architecture developed in the last section the next step is to define the recipe definition and its execution on PLCs. We based our recipe definition on the recommendations NE 033 [10] from the NAMUR association as they define general requirements for recipes. Summarizing these recommendations a recipe controlled automation needs to fulfill the following requirements: recipes should be encoded in a generic way, recipe exchange as well as execution of module functions should be supported, recipes should allow parameterization of module functions, support of conditional execution of commands, iteration of steps should be possible, internal data storage for a module instance should be supported, and support of the application of a module function to a concrete module instance. In addition to these general requirements, we need to consider for our developments that the recipe definition will be stored and executed on a PLC.

Applying these requirements to the domain of discrete automation we identified the following four main elements of our recipe definition:

1. Trigger the execution of a module function (e.g., move cylinder to X, turn on conveyor belt in direction)
2. Conditional execution consisting of an evaluation statement `IF` (e.g., is cylinder in Position X, is workpiece color X, is working area free) and an according stop of the evaluation `END_IF`
3. Indication on the end of the recipe
4. A `GOTO` statement which can be used to create loops

The recipe loaded into a station is stored in the recipe storage which is an array of recipe commands. The structure of the recipe storage and a recipe command is

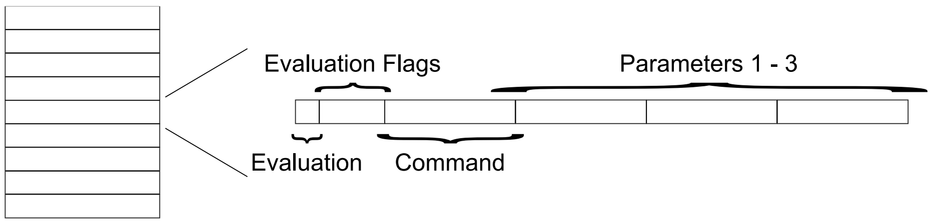


Fig. 3. Composition of Recipe

depicted in Figure 3. Each command consists of a flag used to indicate whether the command is an evaluation used as branch instruction. An array of flags is used to define to which branch a command belongs to. The flags are only used if an evaluation is active (i.e., evaluation command which has not been closed yet). To allow nested evaluations the array of flags is used. The next field specifies the command. The command is specified using the identifier of the module on which the command shall be applied on, the interface identifier, and the command inside the interface. At the end additional parameters can be specified. For the parameter storage we had to take into consideration that IEC 61131-3 only allows fixed sized data structures and arrays as well as PLCs typically have a restricted amount of memory available. Therefore we had to find a trade of for the number of parameters which can be provided directly with a command. In our current implementation we allow to directly provide three parameters with the command. In order to not restrict our design subsequent recipe command entries can be used to store additional parameters if required. These command entries are marked as parameter entries and are handled in the command decoding stage described below.

The recipe execution is carried out in three steps similar to the working model of a CPU [16]. It starts with the fetching of a command. Which is followed by the decoding of the parameters. If a command uses more than three parameters additional command fetches are performed till all parameters could be decoded. After all parameters have been decoded the module identified in the command is selected and the command as well as its parameters are handed over to the respective module for execution. As a command is normally not finished within one cycle of the PLC the recipe execution stays there till the command has been completely executed. This behavior is shown in Figure 4.

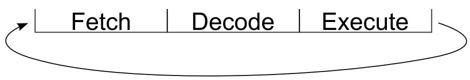


Fig. 4. Execution Steps for Recipe

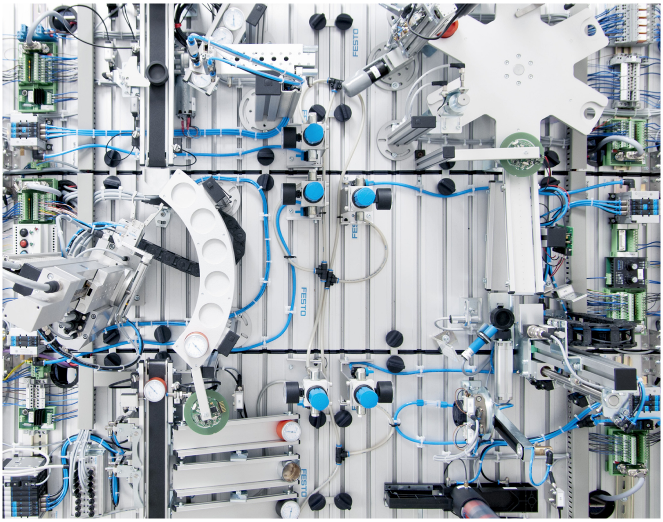


Fig. 5. Setup of the production system. This setup is used to produce black, red, and silver thermometers.

5 Application Example

To illustrate the approach, we used a simplified example from the discrete automation domain used for educational purposes. The setup consists of six different Festo modular production system stations and one conveyor belt as shown in Figure 5. These stations are from bottom right in counter clockwise order: a part supply station, a station for checking supplied parts, a work cell with a drill, an assembly station, a storage for intermediate parts, and finally a sorting station.

The manufacturing system produces temperature sensors in three different colors: black, red, and silver. The different steps required for such a production include distributing of material, testing, processing, assembling, storing, and delivering. An example recipe for the drilling station is for example: wait for part on inlet, turn table one position, check part measurements, turn table one position, if part needs drilling perform drilling operation, turn table one position, eject part, which is the end of the recipe.

We implemented the described recipe-based control system in the S7 PLCs from Siemens, which are controlling the plant. A higher level coordination control provides the recipes for the PLCs and coordinates all stations according to the needs of the different products.

6 Conclusion and Outlook

Production systems need to get more flexible and adapt themselves to different products and product variants with as little human intervention as possible.

With this work we investigated how the flexible recipe-based control can be transferred from batch automation systems to discrete manufacturing. We defined a general control architecture based on modular production cells which will execute product specific recipes. The definition and execution of recipes is based on the experiences gained in the domain of batch automation. By applying this concept to the control system of a demonstration plant we could show the aptness of our approach for increasing the flexibility and adaptability for discrete manufacturing systems.

Based on these results our further investigations will include an evaluation on different kind of processes and to provide module services in a more standardized way to the higher level system. A potential technology can be for example OPC UA. The developed system with its flexibility on the lowest control level can serve as an interesting basis also for highly flexible supervising control strategies like the HMS approach.

References

1. Cyber-Physical Systems: Driving Force for Innovation in Mobility, Health, Energy and Production. acatech Position, acatech – National Academy of Science and Engineering (2011)
2. Arens, D., Hopfgartner, T., Jensen, T., Lamping, M., Pieper, M., Seger, D.: Packaging Machine Language V3.0 Mode & States Definition Document. OMAC Motion for Packaging Working Group (June 2006)
3. Brandl, D.: Design Patterns for Flexible Manufacturing. ISA (2006)
4. Christensen, J.: Holonic manufacturing systems—initial architecture and standards directions. In: 1st European Confernce Holonic Manufacturing Systems (1994)
5. Covanich, W., McFarlane, D.: Comparing the control structures of isa s88- and holonic component-based architecture. IEEE Trans. on Systems, Man, and Cybernetics—Part C: Applications and Reviews 41(1), 4–13 (2011)
6. Eckert, K., Hadlich, T., Frank, T., Fay, A., Diedrich, C., Vogel-Heuser, B.: Design patterns for distributed automation systems with consideration of non-functional requirements. In: 17th IEEE International Conference on Emerging Technologies & Factory Automation (2012)
7. Keddis, N., Kainz, G., Buckl, C., Knoll, A.: Towards adaptable manufacturing systems. In: IEEE International Conference on Industrial Technology (ICIT 2013). IEEE (2013)
8. Koren, Y., Heisel, U., Jovane, F., Moriwaki, T., Pritschow, G., Ulsoy, G., Van Brussel, H.: Reconfigurable manufacturing systems. CIRP Annals-Manufacturing Technology 48(2), 527–540 (1999)
9. Lohse, N., Hirani, H., Ratchev, S., Turitto, M.: An ontology for the definition and validation of assembly processes for evolvable assembly systems. In: The 6th IEEE International Symposium on Assembly and Task Planning: From Nano to Macro Assembly and Manufacturing (ISATP 2005), pp. 242–247 (2005)
10. NAMUR: NE 033 Requirements to be met by Systems for Recipe-Based Operations (January 2003)
11. Sauer, O., Jasperneite, J.: Adaptive information technology in manufacturing. In: CIRP Conference on Manufacturing Systems, Madison, WI, USA (June 2011)

12. Shen, W., Wang, L., Hao, Q.: Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 36(4), 563–577 (2006)
13. Sorouri, M., Patil, S., Vyatkin, V.: Distributed control patterns for intelligent mechatronic systems. In: IEEE Conference on Industrial Informatics (2012)
14. Storr, A., Lewek, J., Lutz, R.: Modeling and reuse of object-oriented machine software. In: Proceedings of the European Conference on Integration in Manufacturing, pp. 475–484 (1997), http://elib.uni-stuttgart.de/opus/volltexte/1999/407
15. Sünder, C., Zoitl, A., Christoph, D.: Functional structure-based modelling of automation systems. International Journal of Manufacturing Research 1(4), 405–420 (2006)
16. Tanenbaum, A.S., Goodman, J.R.: Structured Computer Organization, 4th edn. Prentice Hall PTR, Upper Saddle River (1998)
17. Vogel-Heuser, B., Kegel, G., Bender, K., Wucherer, K.: Global information architecture for industrial automation. ATP (2009)
18. Zäh, M.F., Beetz, M., Shea, K., Reinhart, G., Stursberg, O., Ostgathe, M., Lau, C., Ertelt, C., Pangercic, D., Rühr, T., et al.: An integrated approach to realize the cognitive machine shop. In: Proceedings of the 1st International Workshop on Cognition for Technical Systems, München, Germany (2008)
19. Zha, X.F., Lim, S.Y.E., Lu, W.F.: A knowledge intensive multi-agent system for cooperative/collaborative assembly modeling and process planning. Transactions of the SDPS, Journal of Integrated Design and Process Science 7(1), 99–122 (2003)
20. Zoitl, A., Prähofer, H.: Guidelines and Patterns for Building Hierarchical Automation Solutions in the IEC 61499 Modeling Language. IEEE Trans. on Industrial Informatics PP(99), 1 (2012)

Structural Self-organized Holonic Multi-Agent Manufacturing Systems

José Barbosa[1,3,4], Paulo Leitão[1,2], Emmanuel Adam[3,5], and Damien Trentesaux[3,4]

[1] Polytechnic Institute of Bragança, Campus Sta. Apolónia, Apartado 1134, 5301-857 Bragança, Portugal
[2] LIACC - Artificial Intelligence and Computer Science Laboratory, R. Campo Alegre 102, 4169-007 Porto, Portugal
[3] Univ. Lille Nord de France, F-59000 Lille, France
[4] UVHC, TEMPO research center, F-59313 Valenciennes, France
[5] UVHC, LAMIH, F-59313 Valenciennes, France
{jbarbosa,pleitao}@ipb.pt
{emmanuel.adam,damien.trentesaux}@univ-valenciennes.fr

Abstract. Worldwide systems are increasingly growing into unprecedented complexity levels. This increase of system complexity has to be tackled with new control approaches, where decentralization is playing an important role and particularly the Multi-Agent Systems and Holonic Systems. Despite the benefits of this distribution, new problems arise such as the need for entities coordination. This paper proposes an innovative holonic multi-agent system architecture, named ADACOR^2, which sets foundation on the already proved ADACOR architecture. This new control architecture is empowered by a two vector self-organization, called behavioural and structural self-organization. This paper describes the structural self-organization vector, particularizing the need for embedding in it learning techniques and nervousness stabilizer. A futuristic test bed, inspired in a real flexible-manufacturing system, is used to demonstrate the benefits of this vector in the architecture.

Keywords: self-organization, multi-agent systems, reconfigurable manufacturing control.

1 Introduction

Nowadays, problems' complexity has no parallel in past since the world complexity has grown into unprecedented levels. As examples, in avionics the system complexity has grown from the first airplanes to the nowadays state of the art airplanes, such as the Airbus A380, or in economics where an evolution from local economy to worldwide global markets has noticed. Complexity can also be found on diverse fields of modern society and range from telecommunications to economics or from health to ecosystems in nature. This amazing level of system complexity makes urgent the development of new solutions, methodologies and approaches to better tackle the current demands, such as controlling the chaotic behaviour that these systems may

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 59–70, 2013.

display. Also important to consider is to manage the butterfly effect present in such systems, where small changes in one part of the system can have a huge impact on the other side or at medium/long term. Manufacturing systems domain is also increasingly becoming more complex where a multitude of unexpected or unpredicted events are constantly appearing, such as rush orders, resources malfunctions, late changes requests, order cancelation/modification.

The classical solution to solve these problems passes by the use of information and communication technologies and high processing power, usually using a centralized structure, where decision-making functions are centralized in one node that guarantees high levels of performance optimization. This is currently true for most of the control systems and in fact this approach still works pretty well if one assumes that the system is perfect in the sense that it doesn't have unexpected events. But, the truth is that nowadays industrial systems must deal daily with unexpected events and in that sense must promptly respond, effectively, to them. In this way, these traditional centralized approaches are unable to properly cope with these requirements due to its rigid structure.

One of the most consensual solutions to handle very dynamic and complex systems is the distribution of the processing capacity throughout the system by giving to individuals, more or less intelligent, the capacity to manage its local action space. In this approach, the global system's goals are achieved by cooperation, e.g. by means of information sharing or by the association of individual skills.

Several paradigms have risen to address these challenges, proposing distributed approaches, e.g. Multi-Agent Systems (MAS) and Holonic Systems (HS). MAS [1] derive from the artificial intelligence field and is built upon autonomous and cooperative entities called agents, which are intelligent and autonomous entities capable to take decisions based on local knowledge. HS got inspiration by the work of A. Koestler [2] and has several differences from the MAS such as the creation of stable intermediary states as a mean to achieve optimization or the notion of holarchy which is a mixture of heterarchical autonomous entities (like in MAS) and hierarchical organizations, possibly considering the use of recursivity. The elementary concept in a HS is the term holon, proposed by Koestler, which derives from the Greek word *holos* which means whole and the suffix *on* that suggests a part like in proton. Another key difference between HS and MAS is that a holon can easily be designed as the integration of a physical with an informational part, facilitating the design of more real time and control-based approaches.

Detailing into manufacturing systems, these two paradigms, particularly MAS applications, have been used in the past in several different applications (see for example the surveys [3, 4]). One example, of a Holonic Manufacturing System, is the PROSA reference architecture, which defines the basic holons that a manufacturing system must have as also their basic interactions. Other example is the ADACOR (ADAptive holonic COntrol aRchitecture for distributed manufacturing systems) architecture [5], that proposes an adaptive production control system that switches from a stable state, where the system is organized in a hierarchical approach to achieve optimization, to transient state, where the control is passed into the lower level entities for a prompt response to disturbances. This binary state has shown very good results facing very condition changing environments, but restrings the system to the two pre-defined states.

The current work sets foundation on this already proven holonic system and unleashes it from these two states by allowing the system to dynamically evolve without any pre-defined boundaries. This evolution, called ADACOR^2, is supported by the self-organization concept, found in biology and other fields, decoupled in two different levels: behavioural and structural self-organization. The first one, appears at the micro-level, and is related to the increase of the performance of the individual holons by means of proper adaptation of their behaviour. The second one, described in this paper, appears at the macro-level, and is about the dynamic structural organization of the holons. Having this mechanism in mind, it can be stated that the ADACOR evolved into a more dynamic and open structure allowing the re-arrangement of the entities in the system, exploring the unpredictability of complex systems.

In such dynamic self-organized systems, the problem of nervousness [6–8], which is related to the will that entities have to change at the minimum stimuli from external or internal events, should be considered to guarantee the system stability. Additionally, the development of structural self-organized systems must consider important questions, like what is the best structural configuration, when the system should evolve/re-configure, and how to evolve, which answers may consider the implementation of learning mechanisms.

This paper depicts the architecture for the structural self-organization, as part of the self-organization principle used within the ADACOR^2 approach. To fully demonstrate the proposed approach, the architecture was implemented, tested and validated using a Flexible Manufacturing System (FMS) that is a re-design of the real AIP-PRIMECA, located at the of the Université de Valenciennes et du Hainaut-Cambrésis, in which the resources have moving capabilities.

The rest of the paper is organized as follows: Section 2 overviews the self-organization concept defined in the ADACOR^2 holonic multi-agent system architecture and Section 3 details one of its vectors, the structural self-organization. Section 4 describes the internal architecture of a holon belonging to ADACOR^2 focusing the model to support the self-organization capabilities. Section 5 presents the experimental validation of the proposed approach by using the AIP PRIMECA flexible manufacturing system. At last, Section 6 rounds up the paper with the conclusions.

2 ADACOR^2: Enhanced by Self-organization

The ADACOR control architecture defines four types of holons [5]: Supervisor Holon (SH), Product Holon (PH), Task Holon (TH) and Operational Holon (OH). Besides defining the roles, behaviours and interactions among these holons, ADACOR also introduced an adaptive production control based on balancing the control structure between a stationary state (where the holons are organized according to a hierarchical structure) and a transient state (where the holons are organized in a complete heterarchical structure). The stationary state should be used as much as possible to achieve optimization, and the transient state should be adopted in presence of perturbations to provide adaptive response (returning to the stationary state after the resolution of the perturbation, which should be as fast as possible).

This switching mechanism proved to be very efficient [5] to manage the production control in very unpredictable scenarios, but is restrained to the two pre-defined states. ADACOR^2 unleashes the architecture from these chains by allowing the system to dynamically evolve throughout a set of configurations, dynamically discovered, using the concept of self-organization usually found in biology.

The self-organization mechanism considered in ADACOR^2 comprises two vectors, as illustrated in Fig. 1: the first vector, to cope with smaller impact perturbations, called behavioural self-organization, is achieved by providing holons a set of different behaviours that are dynamically selected accordingly to the current needs. The second vector, called structural self-organization, is more related to dynamic changing of the relations and interactions among holons. Additionally, since ADACOR^2 follows the holonics principles, the structural self-organization is also related to the establishment of intermediary states e.g., the group formation. This macro-level self-organization definition allows the system to dynamically adapt, in a more drastic manner, to larger perturbations by changing the holons relations.

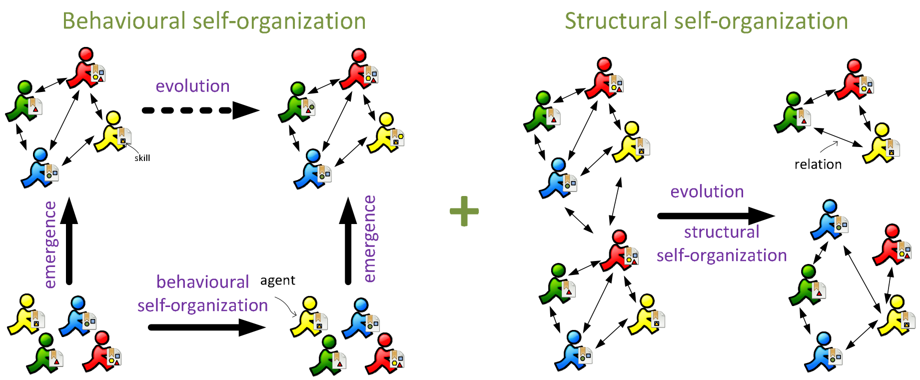


Fig. 1. Self-organization in ADACOR^2 holonic architecture

In this way, a configuration is defined as a snapshot of the system at a given point in time, having information of the holons state, behaviours and their relations (representing the organizational structure of the system). This context-aware knowledge is helpful in situations where the system must evolve, in the way that holons reason on this set of configurations to learn from past situations. To achieve this, ontologies plays an important role enabling the sharing of knowledge in this dynamic and distributed system. In this way, besides the ADACOR manufacturing ontology [9] that has already been developed, a complementary ontology is developed to support the sharing of knowledge related to the configuration contextualization in this dynamic, distributed system. In case of conditions change, the holon(s) that have recognized the situation can find a starting point from similar conditions through the ontology reasoning. This warm, i.e. not being executed from scratch, self-organization allows a faster and most optimized way to evolve into a stable configuration.

The decision of how to better cope with the condition changes, considers learning mechanisms aggregated to a nervousness mechanism to select the best option to take from either behavioural or structural self-organization (pushing the system into the limits but remaining under control). This selection is accomplished by analysing the performance indicators output from the known self-organization mechanisms of the holon. Additionally, learning mechanisms are also responsible to detect new opportunities to evolve, making possible for the system to increase its performance even if on the absence of a disturbing event.

Resuming, the main difference between ADACOR^2 and its predecessor is the removal of the two pre-defined states by introducing the concept of self-organization spited at two levels: behavioural at micro level and structural at macro-level.

Next sections will detail the structural self-organization mechanisms.

3 Structural Self-organization in ADACOR^2

In this work, the structural self-organization is classified in a three level division. This classification closes the gap with the behavioural self-organization by relating it with the structural self-organization.

- Level 0 (emergence): the relations between holons are changed as consequence of the behavioural self-organization. The holon by changing its internal behaviour could, indirectly, imply changing its external relations. This level of structural changing is classified as weak since the structural change is not driven directly by the need.
- Level 1 (logical structural self-organization): each holon is always trying to optimize its place into the holarchy structure. This constant optimization may drive the holon to change from holarchy, to belong to several holarchies at the same time or as freelancer to work completely autonomously.
- Level 2 (physical structural re-organization): similar to level 1 with the addition that holons, e.g. OHs, can physically change their place, changing not only their relations and positions in the holarchy but their physical places.

The stimulus that can trigger a structural self-organization can be any event that disrupts or deviates the predicted function of the system, such as a rush-order, order cancellation, production quality issue, supply shortage, or resource malfunction. To illustrate this, two examples can be provided. The first example of a stimulus that can start a structural self-organization is the introduction of a big batch order. Let's assume that the system is functioning in a stable state with a given configuration, and that at a given point in time, a very large order arrives into the system. The system, realizing these new constraints, will select the most adequate structure to better cope with this. In the simplest way, individual holons can adjust themselves by changing their behaviour or if the impact to the system is large, the holons may find the need to start a structure re-organization. In other words, the holons can change relations keeping the holarchies composition, holons can change holarchies or, if possible, the physical resources can move into another working position to improve the performance indicators.

A second example, at a longer time window, concerns the smoothed changing in a usual set of recurrent orders. Let consider two types of products, A and B. Let also consider that a customer asking for a specific kind of product type B takes more and more importance compared to historical customers asking for product type A. This evolution in the production demand can be detected and the re-arrangement of the structure occurs smoothly to evolve with the needs of customers from a production optimized to make product of type A to a production optimized to make products of type B.

A pertinent question is how does the ADACOR^2 holons self-reorganize structurally? Getting inspiration from the social behaviour of swarms of fishes and birds, it is concluded that they function very well as a group, maintaining system equilibrium and, as a group, avoid predators (in our case, these are the system external perturbations). What is more amazing is that this is achieved by following very simple rules and without any central authority in charge.

ADACOR^2 holons follow the same basic principles as swarms, trying to constantly make cohesive groups but maintaining distances and having crowd management. One observed principle that is not followed in ADCOR^2 is the lack of central authority. In fact, a central authority is used to create stable intermediary states into the holonic holarchy introducing high levels of optimization. These higher level holons, i.e. SHs, have under their supervision a set of OHs, which they try to optimize. Another atomic rule in structural self-organization, under the SH umbrella, is to endow each group with a set of services, as diverse as possible, opening the possibility to attend a wider set of requests. This rule is not restrictive, allowing making more dedicated groups, i.e. more skilled groups, and also makes the system less dependent of a limited group of holons. This characteristic can be seen in Fig. 2, where the services are distributed uniformly among groups.

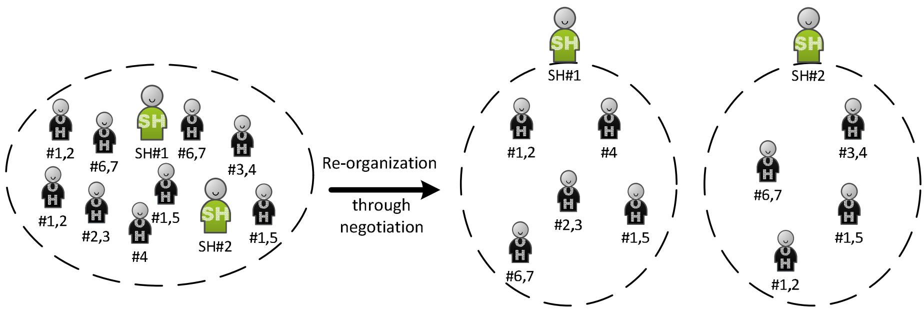


Fig. 2. Equality services distribution in ADACOR^2 group formation

Another crucial rule is related to the possibility of an OH be shared between SHs. Let's suppose that a given service is missing under a stable state governed by a SH. At a given point in time, a SH may find the need to acquire one more service in order to respond accordingly to an external demand or even to the malfunction or decommissioning of the existing OH that offers that service. In this case, the SH will search within the system domain for OHs that offers the same service. Once found, the SH will start a negotiation process, with the other SH or directly with the OH in case of no hierarchical dependency, to find equilibrium between these two groups, i.e. not degrading heavily the service from the source group but allowing the second one to have the possibility to acquire that service. In the case of OH sharing among SHs, its utilization is achieved by the higher level negotiation of the SHs to whom he belongs. An example of this mechanism is depicted in Fig. 3 where the OH that offered the service #4 becomes unavailable, triggering the search to re-acquire the lost service. After a negotiation process (i.e., the first agreement from the SHs level), the SH#1 proposes the OH that has the desired service to belong also to its group. If agree, the holons start the structural re-arrangement and relationship redefinition ending on the holon sharing among groups.

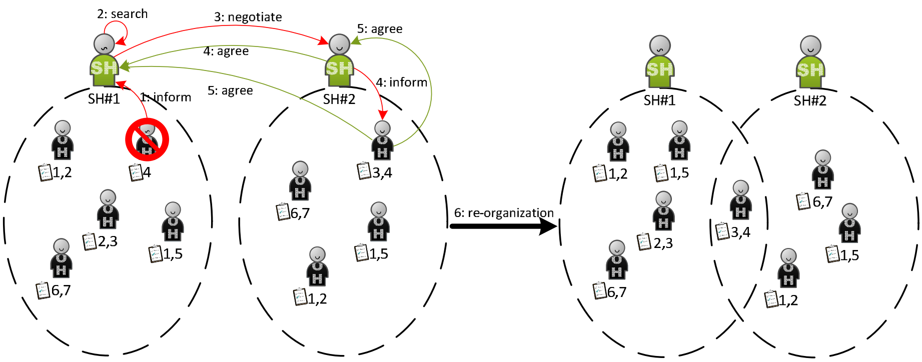


Fig. 3. Structural self-organization initiation: service disruption

Naturally, ADACOR² being a holonic multi-agent system allows that this negotiation process may be started directly with the OH, bypassing the SH.

4 Architecture of Individual Holons to Support Structural Self-organization

The structural self-organization of the system is driven by the behaviour of the distributed holons that constitutes the ADACOR² system. For this purpose, each one exhibits proper mechanisms in its internal structure architecture to support the structural self-organization mechanism. Fig. 4 illustrates the internal architecture of a holon belonging to ADACOR², focusing mainly the model to support self-organization, which comprises primarily the monitoring, discovery, reasoning, nervousness stabilizer and dispatcher components.

The structural self-organization process can be very time-consuming, especially in large-scale systems. In this way, ADACOR² has on the start of this process a context-aware feature. The system, at every disrupting event, takes a snapshot of the system state additionally with the actions taken and the achieved results (i.e., performance indicators). This context-aware mechanism, aided with data processing, e.g. data mining, will facilitate futures events by allowing the system to filter from the best measures taken to similar events and start a warm structural self-organization.

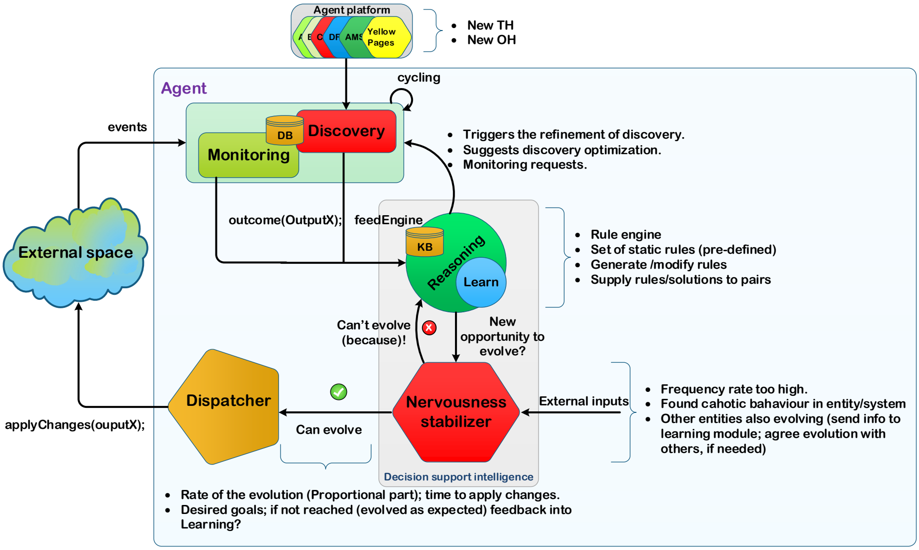


Fig. 4. Model to support the self-organization mechanisms

The discovery and monitoring modules, supported by a knowledge base, will feed the reasoning and learning engines that are responsible to discover new opportunities to evolve and decide how to evolve. In this approach, the learning capabilities embed in holons assume crucial importance to support the generation of new knowledge, contributing to behave better in the future. The implementation of such learning capabilities can range from simple observation or repetition techniques to more complex ones, such as neural networks or support vector machines. The selection of the learning techniques to be deployed in ADACOR^2 must consider several features. Among them are the type of communication to be considered (direct or indirect) and scalability. Other issues are related to the fact that multi-agent systems are typically deployed into dynamic environments and with problem decomposition to reduce complexity [10].

ADADCOR^2 uses different types of learning techniques in different parts of holons or different parts of the system. Social learning is used to support the propagation of new behaviours among the holons or by PHs when creating THs, passing accumulated knowledge from past THs. A reinforcement learning mechanism may be used as a behaviour mechanism for route selection. This reinforcement learning mechanism, inspired in the ant food foraging mechanism, separates the exploration and exploitation phases, and relies in the random decision of the holon as a way to discover new solutions. The feedback from past decisions is also used, where bad past decisions will have a negative impact in the future choice for the same decision as opposite to a good result that will have a positive impact.

In such dynamic environments, where holons are autonomous and self-organized, the system has the predisposition to become unstable and chaotic. Instability and chaos are addressed in ADACOR^2 by applying nervousness stabilizers to each of the two self-organization mechanisms. The purpose of the nervousness stabilizers is to bring stability into the behaviour of the holons and to the structural self-organization in ADACOR^2. By introducing these stabilizers, the system operates in a such manner that is pushed to its limits by enhancing the self-organization and chaotic principles, but always under control. Note that any decision to evolve provided by the reasoning and learning engines is evaluated in the nervousness stabilizer which will validate the decision.

Typical approaches to calm the holons' will of change passes by restring the number of changes during a given time, by allowing the entities to change only at predefined intervals or by setting the exploration/exploitation threshold [8]. ADACOR^2 holons use an innovative technique [8] where the inspiration from the classical control theory, namely the Proportional, Integrative and Derivative (PID) controller, acts as a constrain to the response time and defines the accepted deviation from the known optimal plan or the speed of the deviation mitigation.

5 The AIP Cell Application Example

The testing and validation of the proposed structural self-organization mechanism uses an adapted version of the AIP-PRIMECA manufacturing system, located at the Université de Valenciennes et du Hainaut-Cambrésis. In this scenario, resources can self-reorganize, responding in a more appropriate way to perturbations maintaining high levels of performance. This test scenario allows to verify all potentialities of the structural self-organization of ADACOR^2.

5.1 System Description

The real AIP-PRIMECA manufacturing system comprises a set of 6 resources, each one able to perform a set of services that are linked by a fixed conveying system [11].

The proposed system assumes that resources have moving capabilities in the sense that they can unplug from one working position and be moved into another working position. A working position is a place in the environment that is equipped with supporting systems for resource normal functioning, such as power and communications. The rail type conveyor system present in the real AIP cell is removed and the transportation between resources is accomplished by using Automatic Guided Vehicles (AGVs). Fig. 5 depicts the conceptual view of the futuristic cell, where the purple places represent working positions and the green ones can only be used for routing paths. For transportation calculation times it is assumed that moving from one point to another takes 3 seconds to the AGV in a clear path. The shifting of resources depends from resource to resource since the complexity of cables, size and weight can vary.

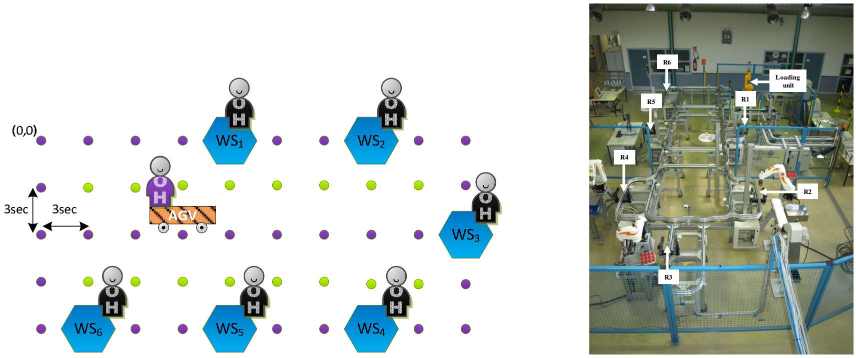


Fig. 5. Cell configuration test scenario (on the rigth side the real AIP cell)

Having this in mind, the ADACOR[2] holons are deployed into the production control system, with resources being mapped as OHs and each AGV as a specialized transporter instantiation of the OH (see Fig. 5). This test bed is composed by 6 resources, each one having a number of offered services, as illustrated in Table 1.

Table 1. Resource service list and processing times

service	M_1	M_2	M_3	M_4	M_5	M_6
Loading	10					
Unloading	10					
Axis		20	20			
R_comp		20	20	20		
I_comp				20		
L_comp		20	20	20		
Screw_comp						
Inspection					5	
Recovery						60

A level 3 structural self-organization was used for the tests, since resources will have the possibility to shift place in the shop-floor. The structural self-organization mechanism is triggered when a batch of orders is allocated to an OH. After receiving a batch of orders, the OH will start an information gathering from other OHs, where resource queue, allocated work-orders, processing times and actual location, is exchanged. The procedure used follows the principle that the most overloaded OH are the most critical ones and in that way tries to allocate them first by minimizing the transportation times between them. After finishing the allocation of all OHs, each OH sends the information of either better or worst solution achieved, from actual resources disposition. In the case of a better solution, the OH sends the KPI and the new allocation places to all the OH. The overall best solution, found in each OH, is automatically assumed and used.

5.2 Analysis of the Results

The described system, developed using the JADE framework [12] and the Java Expert System Shell [13], initializes with a manufacturing order to produce 2xBELT. After 120 seconds a new manufacturing order to produce 5xAIP appears in the system, which triggers a structural self-organization, see Fig. 6.

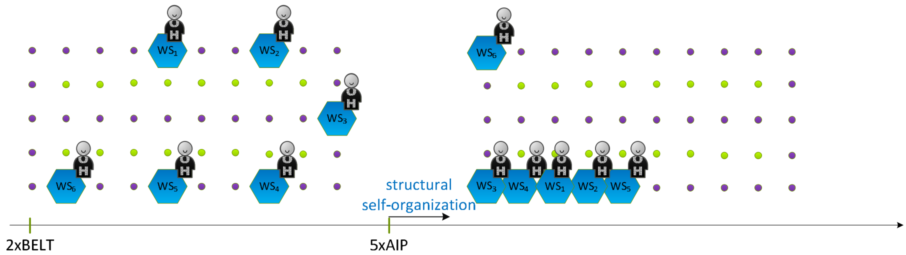


Fig. 6. Structural self-organization in practice

Two scenarios are simulated and compared to evaluate the structural self-organization. On the first scenario the structural self-organization is disabled while it's enabled on the second one. For each scenario, several simulations were done and average values are presented in Table 2.

Table 2. Experimental results

	Without structural self-organization	With structural self-organization
C_{max}	1147,5 s	1112 s
%transpTime	19,4 %	14,9 %
avg(order)	725,2 s	646,2 s

As can be observed by the analysis of the previous table, enabling the structural self-organization reduces the overall Cmax, allowing a performance improvement. This is mainly accomplished by the reduction of transportation times when the structural self-organization is enabled (reduction from 19,4% to 14,9%). Also a reduction of the time to process each individual order was noticed.

The achieved preliminary experimental results show the merits of the structural self-organization approach to face severe changing conditions. However, the experiments also showed new possibilities to improve the proposed algorithm, e.g. by considering optimization in the re-configuration of the resources during the structural self-organization process.

6 Conclusions

System complexity has grown into unprecedented levels, turning on the need of new control architectures. In this subject, MAS and HS play an important role by proposing a distribution of the processing capacity by several autonomous entities.

This paper introduced an innovative control architecture called ADACOR^2, which considers a two vector self-organization mechanism, named behavioural and structural self-organization. The structural self-organization, detailed in this paper, operates in a macro-level and is related to the relations between holons and holarchies formation.

ADACOR^2 by being proposed to work in highly dynamic systems is supported with learning mechanisms allowing the dynamic discover of new ways to evolve. Additionally, by giving autonomy to these entities, they can display some nervousness, making the system to become more unstable. The idea is to push the system to its limits, by enhancing the self-organization and chaotic principles, but maintaining the system under control, by actuating in the nervousness control.

The structural self-organization mechanism was tested in a production cell, inspired in the real AIP cell, where resources have the possibility to shift their position in the shop-floor. The achieved preliminary results show that enabling the structural self-organization mechanisms a performance increase is achieved.

Future work will be devoted to improve the algorithms considered in the structural self-organization mechanism and to merge the two self-organization principals in ADACOR^2.

References

1. Wooldridge, M.J.: An introduction to multiagent systems. John Wiley & Sons, Chichester (2009)
2. Koestler, A.: The Ghost in the Machine. Arkana Books (1969)
3. Leitão, P.: Agent-based Distributed Manufacturing Control: A State-of-the-art Survey. Proceedings of the International Journal of Engineering Applications of Artificial Intelligence 22(7), 979–991 (2009)
4. Leitão, P., Marik, V., Vrba, P.: Past, Present, and Future of Industrial Agent Applications. IEEE Transactions on Industrial Informatics, 1 (2012)
5. Leitão, P., Restivo, F.: ADACOR: A Holonic Architecture for Agile and Adaptive Manufacturing Control. Computers in Industry 57(2), 121–130 (2006)
6. Parunak, H. Van Dyke, Brueckner, S.A., Matthews, R., Sauter, J.: How to calm hyperactive agents. In: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2003, Melbourne, Australia, pp. 1092–1093. ACM, New York (2003)
7. Hadeli, Valckenaers, P., Verstraete, P., Germain, B.S., Van Brussel, H.: A Study of System Nervousness in Multi-agent Manufacturing Control System. In: Brueckner, S.A., Di Marzo Serugendo, G., Hales, D., Zambonelli, F. (eds.) ESOA 2005. LNCS (LNAI), vol. 3910, pp. 232–243. Springer, Heidelberg (2006)
8. Barbosa, J., Leitão, P., Adam, E., Trentesaux, D.: Nervousness in Dynamic Self-organized Holonic Multi-agent Systems. In: Pérez, J.B., et al. (eds.) Highlights on PAAMS. AISC, vol. 156, pp. 9–17. Springer, Heidelberg (2012)
9. Borgo, S., Leitão, P.: Foundations for a Core Ontology of Manufacturing. In: Sharman, et al. (eds.) Ontologies, pp. 751–775. Springer, US
10. Panait, L., Luke, S.: Cooperative Multi-Agent Learning: The State of the Art. Autonomous Agents and Multi-Agent Systems 11, 387–434 (2005)
11. Pach, C., Bekrar, A., Zbib, N., Sallez, Y., Trentesaux, D.: An effective potential field approach to FMS holonic heterarchical control. Control Engineering Practice 20, 1293–1309 (2012)
12. Bellifemine, F.L., Caire, G., Greenwood, D.: Developing multi-agent systems with JADE. John Wiley, Chichester (2007)
13. Friedman-Hall, E.: Jess in action. Manning, Pearson Education, Greenwich, Conn. (2003)

Deployment of Multiagent Mechatronic Systems

Luis Ribeiro and José Barata

CTS, UNINOVA, Dep. De Eng Electrotécnica, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, 2819-516 Caparica Portugal
{ldr,jab}@uninova.pt

Abstract. Multiagent system's applications in industry have been widely investigated to support emerging production paradigms and models. One of the key concepts of these approaches is the utilization of a highly dynamic environment whereby intelligent agents are able to take autonomous decisions and self-organize to circumvent production disturbances. Deployment is a fundamental activity in these systems and entails significant technical and conceptual challenges. There are only a few documented deployment platforms for agents in a mechatronic context. This paper presents the deployment philosophy developed and tested under the framework of the FP7 IDEAS project.

Keywords: deployment, multiagent systems, mechatronic agents, agent-enabled controllers.

1 Introduction

Deployment is a fundamental activity in modern production systems. Normally it implies that executable code is compiled, to generate the respective binary files, in some development computing device and subsequently transferred to a specific controller (target) where those binaries will be executed. In current shop floors this often entails stopping all the activities being supervised by the target. Emerging production paradigms such as Bionic Manufacturing Systems (BMS) [1], Holonic Manufacturing System (HMS) [2] or Evolvable Assembly Systems (EAS) [3]; promote control methodologies that render current deployment practices impractical.

In particular, these approaches envision system modifications as a key activity and enabler of shop-floor agility and consider plug and produce an essential feature. Deployment is closely linked with the plug and produce concept and is probably the first step required to attain a pluggable shop-floor.

Most of the before mentioned paradigms have elected the concept of agent as a mean to instantiate their conceptual framework to a point where multiagent technology itself is sometimes mistakenly confused as a production paradigm itself. An extensive review of agent applications in industry can be found in [4-6].

The currently available technology renders the deployment of multiagent systems (MAS) particularly challenging as it does not offer, alone, the mechatronic deployment models. Further, the relatively high abstraction level required from the programming languages used to implement agent stacks renders their native utilization impossible in most industrial controllers for at least three main reasons:

V. Mařík, J.L. Martinez Lastra, and P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 71–82, 2013.

1. The binaries memory footprint far exceeds the available memory in the controller.
2. The source code and the required framework support are unable to handle hard real time requirements.
3. The processing capabilities of the controller cannot handle the binaries in a suitable time, even to ensure soft real time performance.

It is therefore not a surprise that most researchers and practitioners choose to separate the agent's binary from the hard real time native binary of the controller. Virtually all deployment approaches support a hardware abstraction layer that handles this separation and ensures the communication between the agent and the native code (Fig. 1).

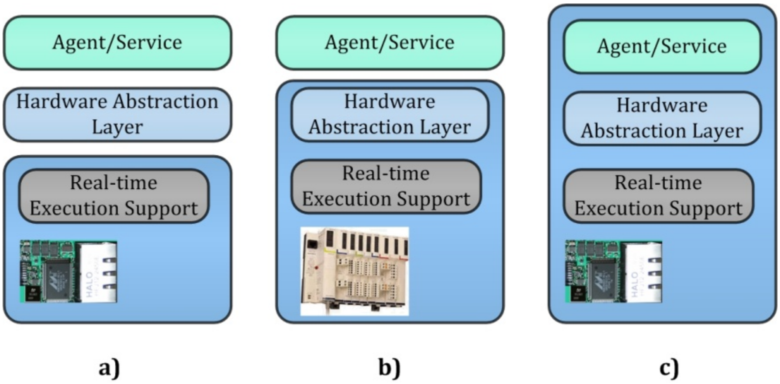


Fig. 1. Approaches to agent and native controller code separation

The second differentiation point between the existing approaches is how much executable data is actually stored in the controller. As mentioned in [7] physically separating the agent binaries from the native binaries (Fig. 1 – a,b) may compromise the performance of the whole since data from the agent must be sent over a network and trigger the real time control events in the controller.

The main purpose of the hardware abstraction layer is to isolate the agent part of the system from specific hardware implementation details so that the agent logic remains universally applicable. In this context, this layer may be either fully implemented in the controller's native language or shared between the agent development language and the native language (which reduces the agent platform independence from the hardware details).

There is a conceptual advantage of considering the scenario of Fig. 1 - c in a mechatronic context. If there is an identity relation between a specific physical device and an agent, mediated by the controller, it means that this tuple (agent, controller, device), known as Mechatronic Agent (MA) in the EAS terminology can be plugged and unplugged as a completely self-contained entity. This identity is of extreme relevance in the before mentioned paradigms. There are not many consistently reported cases of deployment infrastructures in the literature.

One of the most important cases, using agent technology, is reported in [8, 9] and supports the deployment of Holons in Rockwell Automation Controllers. It makes used of shared memory spaces that enable the agent binaries to seamlessly interface with the native binaries while seating in the same controller. Further it uses the concept of Holonic template as a mechanism to design and deploy the holons on the controllers. A Holonic template includes both the information of the native and the agent binaries. In the agent domain the abstraction layer approach has also been applied in the AMES agent stack [10] as a mean to separate the hardware from the agent's behaviours.

These deployment approaches are not exclusive from the agent domain. Recently, Service Oriented Architectures (SOA) and related technologies have followed the same principles. The author's have discussed elsewhere [11] the advantages of merging concepts and technologies to improve automation technology. In this context, the deployment challenges are virtually the same. Examples of deployment of web services in industrial equipment include the work carried out by Schneider Electric (SE) in the projects SIRENA[12] and SODA[13] where the abstraction layer approach was used to build service oriented systems using the Devices Profile for Web Services (DPWS) in SE's and ININCO Technologies controllers that support browser based deployment and configuration of services.

The work reported in this paper was developed under the FP7 IDEAS Project (featured as an FP7 success story) and documents the deployment approach supported by the IDEAS Agent Development Environment (IADE) [14, 15]. The subsequent details are organized as follows: section 2 briefly details the IADE architecture, section 3 presents and discusses the deployment mechanisms, section 4 discusses the main technical details supporting them and finally section 5 concludes the discussion.

2 IDEAS Agent Development Environment (IADE)

The IADE stack is composed of nine main agents as detailed in **Table 1**. These agents are generic and fit specific purposes ranging from platform support, as is the case of Yellow Page and Deployment agents (the latter is the main technical concept discussed). The agent stack is controlled by set of complementary tools (Fig. 2) that handle visualization, agent design, process design and finally deployment. Overall, the user defines as set of AutomationML files (using the MASCOTT tool) that contain the main mechatronic restrictions of the system. These files are them imported to the Agent Configuration tool that converts the information into IADE mechatronic agents (RA's, TEA's, HUA's, etc) and prepares them for deployment. The process design tool is responsible for the design of processes that may require composed functionality to be implemented and supported by CLA's. As the offline design process progresses IADE agents can be added or removed from the system. This action is meditated by the Deployment agent. This architecture has currently been preliminary validated in an industrial setup as detailed in[15] and more tests in distinct industrial scenarios are currently ongoing validation.

Table 1. Main Agents in IADE

Role	Class	Function
Platform Support	Deployment Agent (DA)	Mediates the agent platform and the line controllers and enables the user to deploy any agent from the Process Execution and Transport Roles into a specific controller.
	Yellow Pages Agent (YPA)	Keeps track of the services/Skills each agent can offer in the system. Agents can query the YPA to locate other agents and their skills and to subscribe a notification service that informs them if a specific agent has left the platform.
Process Execution	Resource Agent (RA)	RA's interact directly with the line controllers and represent the lowest possible abstraction entity in the IADE stack. Their main role is to translate skills to native code and ensure its execution and synchronization with the agent platform. RA's inform the transport system about the physical location where their skills can be consumed.
	Coalition Leader Agent (CLA)	CLA's are used to compose functionality at the cost of existing agents. CLA's do not interface with line controllers yet are able to coordinate the execution of other RA's and CLA's. CLA's directly implement sub processes that will be later consumed by Product Agents when executing their process plan. CLA's inform the transport system about the physical location where their skills can be consumed.
	Product Agent (PA)	PA's represent the highest abstraction level in the system and they entail a direct mapping between the agent and a specific item being produced. Each PA has the ability to manage its own processing plan and take supported decisions on the location where each step of the process is executed. The collective action of PA's and the agents of the transport system is the main topic investigated in this document.
Transport	Product Source Agent (PSoA)	The PSoA manages the pallets entering the system and associates a specific pallet with a PA. This association is fundamental since there can be more PA's than available physical carriers (pallets) and in this context the PSoA manages the entrance of PA's in wait.
	Product Sink Agent (PSiA)	The PSiA work in the opposite way of the PSoA. In this context the PSiA releases the pallets from PA's whenever a specific PA terminates its process plan.
	Transport Entity Agent (TEA)	Abstracts a specific conveyor in the system. It is responsible for carrying pallets between diverters. Each TEA hosts several docking points where stations abstracted by RA's or CLA's can be plugged and unplugged in runtime. The TEA is sensitive to these actions and manages all the pallet traffic in it by informing PA's about routing costs and their current position upon reaching their destination.
	Handover Unit Agent (HUA)	HUA's control the system's diverters and continuously compute the cost spanning tree to reach any possible destination in the line. This is information is shared with the associated TEA's which when necessary convey it to PA's.

As depicted in Fig. 2, the DA is responsible for processing all the agent description files generated by the tools and ensures the agent instantiation in any controller of the system.

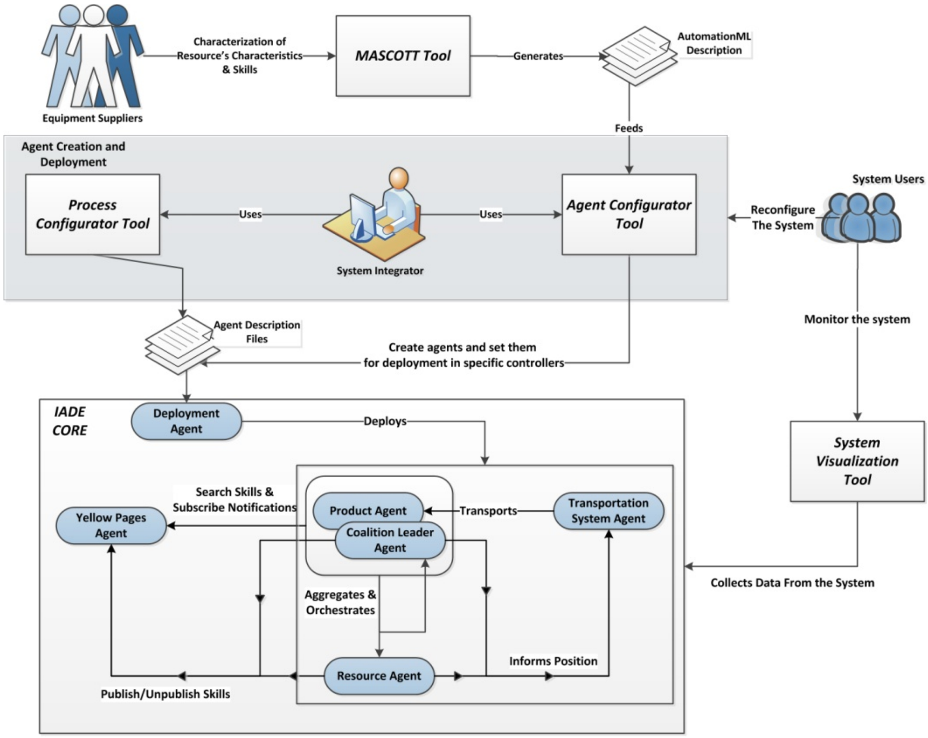


Fig. 2. IADE core and main functional relations between agents and tools

The agent description files encapsulate all the required information to deploy an agent including the references for hardware controller libraries, and their mapping to the agent functionalities, in the case of RA's as well as the process sequence for CLA's and PA's which may include static restrictions that oblige the execution of certain skills by specific RA's or CLA's (other than these, the agents negotiate stepwisely the execution of their process plan). Finally the description files also include the coupling point at which RA's and CLA's can be associated with TEA's.

3 Deploying Agents in IADE

As mentioned before IADE makes use of deployment agents as an entry point, to the system controllers, to enable the deployment of control and transport agents. The deployment agent is bootstrapped with the controller and advertises itself in the agent platform. Since this work uses JADE as the base agent platform this registry is supported by the Directory Facilitator Agent (DFA). DA's are therefore the very first agents to be started in the platform and virtualize the system controllers. The bootstrapping process of a DA is depicted in Fig. 3.

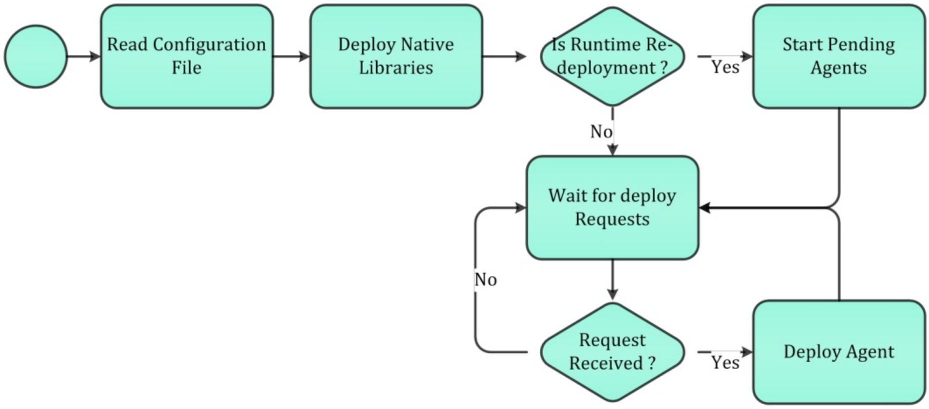


Fig. 3. DA Bootstrapping process

The process starts with the DA processing its local configuration file. This file defines the physical location of the controller on the system and all the native libraries that must be loaded so that IADE agents can be deployed and control that specific controller. This includes the instantiation native I/O mapping as part of the hardware abstraction layer. The next step is the verification of whether this is a redeployment action. The DA's are prepared to be shutdown along with the controller and corresponding module and be deployed in a distinct location in the system. Whenever this occurs the DA creates and saves a listing of all the agents currently running in the controller and properly shuts them down for redeployment. These interactions follow a FIPA Protocol Request pattern (Fig. 4). The redeployment protocol supports the removal of an entire mechatronic agent (Agent + Controller + Device) and its association in a different point on the line. It requires, as depicted in Fig. 4, a chain of interactions normally started by the user through the configuration tool. The user will send a redeploy message to the DA with an indication of the next physical area where the mechatronic agent is to be deployed. This information is fundamental to ensure that upon start (Fig. 3) the DA is able to convey that data to the mechatronic agent so that, in term, it associates itself with a new TEA. These, along with the HUA's, are responsible for managing the proper routing of PA's in the system.

The regular deployment process follows a similar pattern. The agent intercepts them when is in the state of waiting for deployment request. The actual interactions are detailed in Fig. 5. Upon receiving the deployment request the DA de-serializes the agent description and identifies its main class. An instance of that specific class is then created and deployed on the controller. As soon as a mechatronic agent is deployed it informs the local TEA where it has been coupled so that the transport system agents take notice of the new skills in that location and propagate the information. The agents that are technically more complex to deploy are those that directly interface with the system controllers (HUA's, TEA's and RA's).

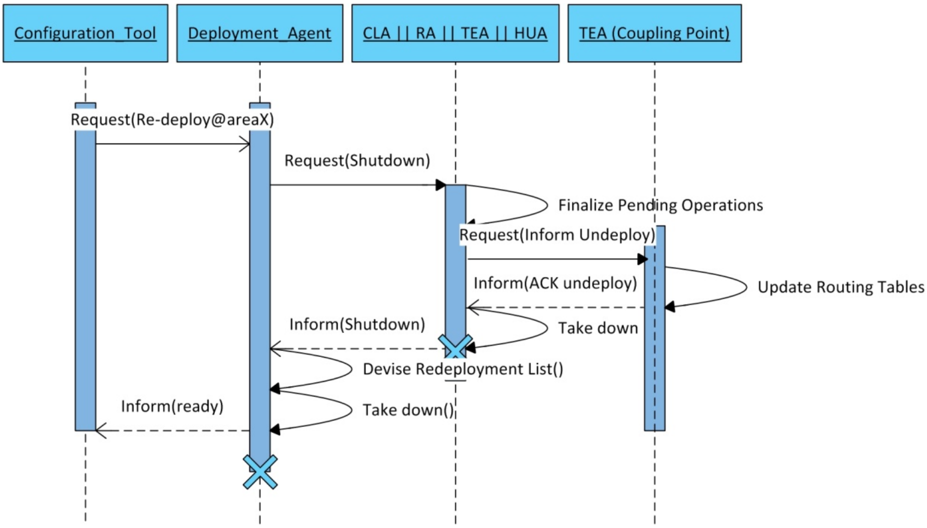


Fig. 4. The Redeployment Process

The successful deployment of these agents also includes the deployment in the target of lower level libraries that implement, through the controller's native binaries, the skills of these agents. The agents dynamically load these libraries in runtime and proceed with the invocation.

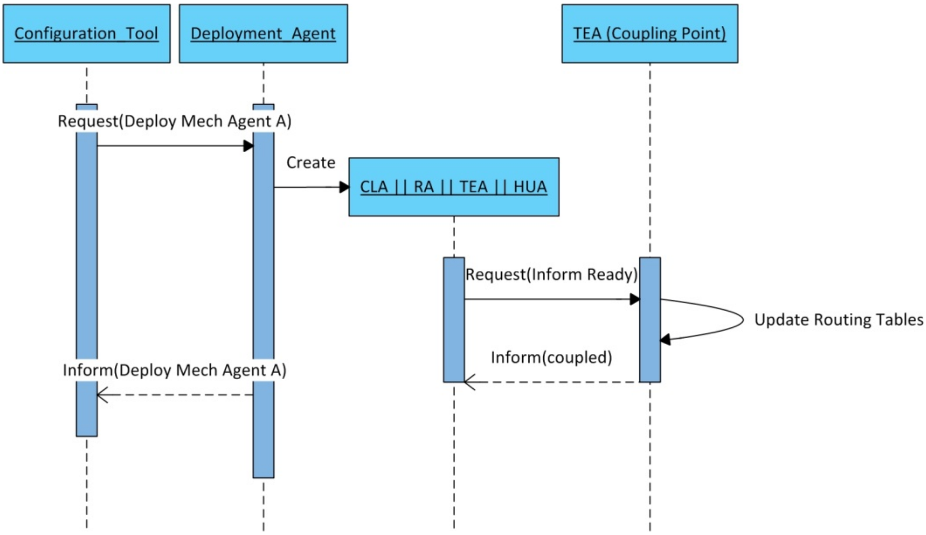


Fig. 5. The deployment process

As discussed later, one of the advantages of this approach is that it renders possible the replacement of the library containing the controller's details without affecting the agent's skills or its location which also means that one agent can be migrated from one controller to another provided that there is a low level implementation of the agent skills which the agent will be capable of managing autonomously without the need for further reprogramming.

4 Technical Discussion

The deployment functionalities earlier detailed make use of unique capabilities associated with the JAVA programming language and supported by any java virtual machine (JVM) compliant with the standard or micro editions of the JVM. The presented architecture has been implemented in using JADE [16] which can also be deployed in both editions of the JVM using JADE-LEAP [16].

The authors shall now analyze the deployment of RA's (deployment of TEA's and HUA's is similar from a technical point of view).

Fig. 6 details a simplified description of the IADE data model which focuses on the skill details. Each RA is an instance of an abstract class (Mechatronic Agent) that can host skills.

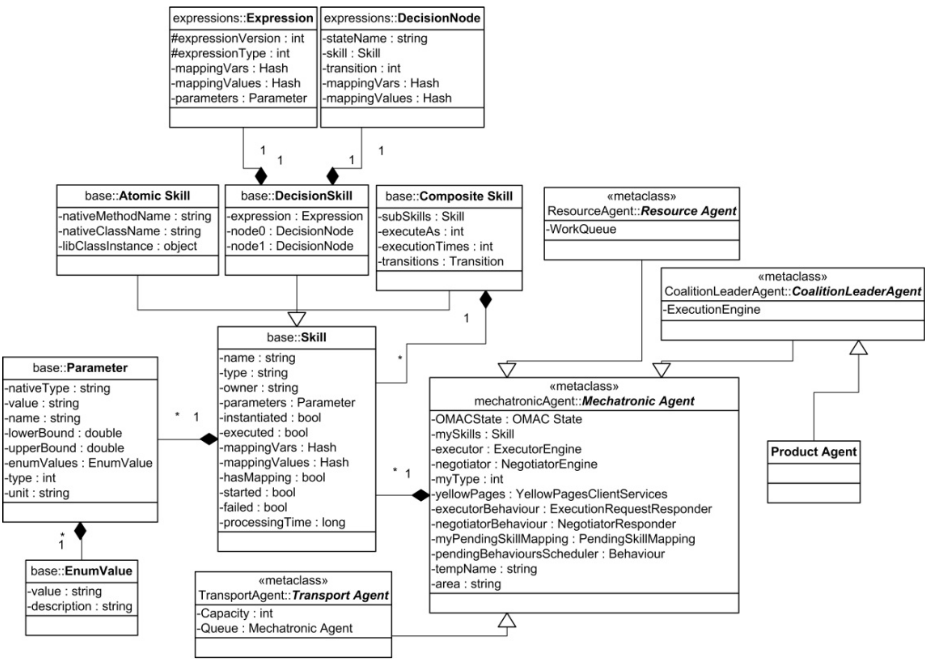


Fig. 6. Description of the IADE data model focusing on the skill definitions from [14]

An RA hosts Atomic Skills. What differentiates such a skill from others is that it contains three pointers: one that encapsulates the name of the method that should be invoked to trigger the execution of the native binaries, a second that contains the name of the library that should be invoked and the third that contains an instance of that library. The behaviour of the RA upon deployment is detailed in Fig. 7.

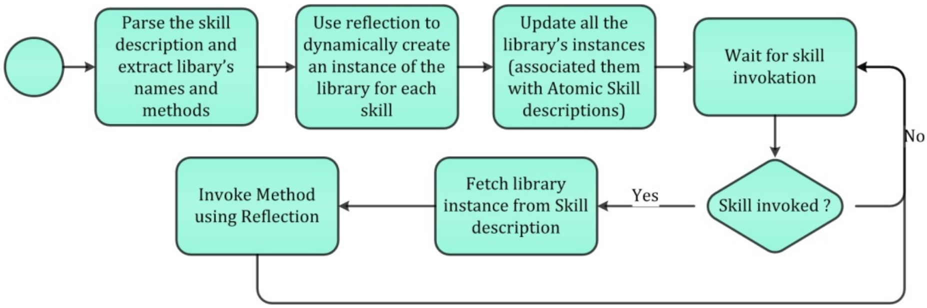


Fig. 7. Runtime management of Atomic Skills and low level libraries

The actions described in Fig. 7 heavily rely in the JAVA Reflection API to dynamically manipulate the low level libraries. In particular, the method invocation state makes use of the parameters described in the skill description (Fig. 6) to match the methods signatures in the low level library. These intermediate java libraries actuate as IADE's hardware abstraction layer. To ensure that IADE can interface seamlessly with any hardware, provided that the harmonizing library exists, IADE has an abstract class that should be extended by that library. Among other things this class enforces that the methods to be paired with the skill should have a signature compatible with the skill's parameters and whose first parameter should be an Atomic Skill.

There are also some rules that should be followed to harmonize the agent execution logic with the native execution. One important detail regarding JADE's behaviour scheduler is that it supports a non-preemptive cooperative logic. In this context, behaviours explicitly release the control over the main thread so that others can execute. IADE preserves this logic. This carries an important consequence. An IADE agent is implemented by several behaviours. The execution engine (Fig. 6) is only one of these behaviours, among others, that ensure the mechatronic agent integrity. This implies that the agent should not be blocked by a time consuming execution. In this context, the agent will attempt to execute as much as possible from the method implementing the low level interaction but will never wait (if the library is properly implemented). The execution engine will then release the control to other behaviours and in the next opportunity will re-entry the method at the point where it was left. In order to do so successfully the method should be implemented as a non blocking finite state machine and should signal the execution engine when the execution of the atomic skill has been completed.

In the text below there is a listing of a compliant library code in JAVA. The class starts by extending the IADESkillImplementationLibrarySkeleton which, as detailed before, ensures the proper method invocation.

```
public class BlankExample extends IADESkillImplementationLibrarySkeleton
{

    private static final int INITIALIZE = 0;
    private static final int POSITION = 1;
    private static final int VERIFY_POS = 2;

    private int state = INITIALIZE;

    public void positionAndExecute(AtomicSkill ask, int position, int
executionCode) {
        switch (state) {
            case INITIALIZE:
                //call initialization routine
                state = POSITION;
                break;
            case POSITION:
                //call positioning routine
                state = VERIFY_POS;
                break;
            case VERIFY_POS:
                if (positionReached()) {
                    ask.setExecuted(true);
                    state = POSITION;
                }
                break;
        }
    }
}
```

In this case the skill description would have a skill named **positionAndExecute** with two integer parameters (the position and the execution code). The library's method follows this signature and adds the Atomic Skill variable as first parameter. This variable is extremely important to interface information between the hardware and the agent /skill logic. For instance, in the example above the executed flag is set to true to tell the agent that the skill execution is finished and that a new skill can be executed. The library does not block the agent in any state. The ask variable, as an instance of a skill (Fig. 6), also enables the harmonization library to read and write skill parameters by channeling the result of certain parameters to other by using the "mapping values" and "mapping vars" tables.

Since the agent only requires the references in the Atomic skill description to trigger all this generic execution logic it is possible to replace the skill implementation without modifying the agent's skill's description. This is of paramount importance since it is one of the key enablers of the advanced deployment functionalities in IADE. It allows:

- Migrating and agent between controllers and making sure that that agent can only execute if a compliant library is present.
- Changing the physical/controller setup without affecting the perception that the agent and other agents have on the supported skills.
- Customizing native code for any platform provided that JAVA support is enabled.

5 Conclusion

The present document details the deployment approach followed under the scope of the FP7 IDEAS project and implemented in the IDEAS Agent Development Environment. Although the deployment functionalities only account for a small portion of the agent stack they are fundamental to ensure the consistent behaviour of the system as a whole. The approach described has the shortcoming of heavily relying in JAVA technology yet it has the virtue of working in any JAVA enabled controller. Although these controllers are currently not standard automation technology and a few where developed under the IDEAS project itself, the paper also serves the purpose of highlighting that certain features that are specific of emerging production paradigms may require more suitable description and implementation languages and infrastructures. In particular the authors' find particularly important the possibility of hot swapping mechatronic agents (Agent + Controller + Device) in runtime in a production context as a measure to boost shop floor agility and to truly explore the very best of emerging conceptual production approaches. The paper discusses the technical details of a mean to do so which, far from being the only reported deployment platform in the literature, has been proven in several industrial demonstrators and in a wide range of platforms from standard pc's, to industrial pc's and controllers supporting different JAVA profiles.

References

1. Ueda, K.: A concept for bionic manufacturing systems based on DNA-type information. In: PROLAMAT. IFIP, Tokyo (1992)
2. Van Brussel, H., et al.: Reference architecture for holonic manufacturing systems: PROSA. Computers in Industry 37(3), 255–274 (1998)
3. Ribeiro, L., et al.: Evolvable Production Systems: An Integrated View on Recent Developments. In: 6th International Conference on Digital Enterprise Technology. IEEE, Hong Kong (2009)
4. Leitão, P.: Agent-based distributed manufacturing control: A state-of-the-art survey. Engineering Applications of Artificial Intelligence 22(7), 979–991 (2009)
5. Monostori, L., Váncza, J., Kumara, S.R.T.: Agent-Based Systems for Manufacturing. CIRP Annals - Manufacturing Technology 55(2), 697–720 (2006)
6. Marík, V., Lazanský, J.: Industrial applications of agent technologies. Control Engineering Practice 15(11), 1364–1380 (2007)
7. Marík, V., et al.: Rockwell automation agents for manufacturing. In: Fourth International Joint Conference on Autonomous Agents and Multiagent Systems. ACM, The Netherlands (2005)

8. Vrba, P., et al.: Rockwell Automation's Holonic and Multiagent Control Systems Compendium. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 41(1), 14–30 (2011)
9. Tichý, P., et al.: Multi-agent system design and integration via agent development environment. Engineering Applications of Artificial Intelligence 25(4), 846–852 (2012)
10. Theiss, S., Vasyutynskyy, V., Kabitzsch, K.: Software agents in industry: A customized framework in theory and praxis. IEEE Transactions on Industrial Informatics 5(2), 147–156 (2009)
11. Ribeiro, L., Barata, J., Mendes, P.: MAS and SOA: Complementary Automation Paradigms. In: Azevedo, A. (ed.) Innovation in Manufacturing Networks. IFIP, vol. 266, pp. 259–268. Springer, Boston (2008)
12. SIRENA. Service Infrastructure for Real-time Embedded Network Applications (2003-2005), http://www.sirena-itea.org/
13. SODA. Service Oriented Device and Delivery Architecture (2006-2008), http://www.soda-itea.org/
14. Ribeiro, L., et al.: IADE – IDEAS Agent Development Environment: Lessons Learned and Research Directions. In: CIRP Conference on Assembly Technologies and Systems (CATS 2012), Ann Arbor (2012)
15. Ribeiro, L., et al.: Self-organization in Automation - The IDEAS Pre-Demonstrator. In: 37th Annual Conference of the IEEE Industrial Electronics Society (IECON 2011). IEEE, Melbourne (2011)
16. JADE. Java Agent Development Framework (2013), http://www.jade.tilab.com

Agent-Based Control of Operational Conditions for Smart Factories: The Peak Load Management Scenario

Anna Florea, Juha Lauttamus, Corina Postelnicu, and Jose L. Martinez Lastra

Tampere University of Technology, Tampere, Finland
{anna.florea,corina.postelnicu,jose.lastra}@tut.fi

Abstract. Operational conditions define the minimum requirements for the manufacturing site to operate while at the same time restricting the optimization potential and limiting the opportunities for reduction of resource consumption. Parameters of the manufacturing ecosystem composing these conditions depend on numerous factors. High-level intelligent control is required in order to grasp the complexity of the dependencies between the manufacturing ecosystem parameters and expand opportunities for optimization.

This paper describes an agent-based peak load management scenario and considers its implication for control architecture of smart factory operational conditions leveraging on SOA and collaborative intelligent agents paradigms.

Keywords: agent-based control, SOA, energy efficiency, peak load management.

1 Introduction

Manufacturing industry is an important part of the modern economy: according to the World Bank data the value added by the sector contributed to 16.6% of global GDP in the year 2010 [1]. As a consequence, it has strong environmental impact accounting for one third of global CO2 emissions and constituting one third of world's energy demand [2]. Even though the problem is being actively studied and new solutions targeting decrease in material and energy consumption are being developed, the report of International Energy Agency (IEA) from 2007 [3] states that the average annual rate of energy savings over the last 20 years has reduced by half in comparison to the period of 1973-1990. In this context information and communication technologies (ICT) have gained attention as important means to improve energy efficiency of the sector.

Over the previous decade ICT provided the manufacturing sector with tools implementing flexible manufacturing systems, much needed to address the requirements imposed by mass customization. The advent of Service Oriented Architecture (SOA) deployed by Semantic Web Services (SWS) supported creation of reconfigurable manufacturing systems relying on field devices provided with SWS-middleware [4], [5]. Further adoption of SOA and EDA (Event Driven Architecture) at the lower levels of automation pyramid enabled vertical integration of automation solutions and opened opportunities for implementation of enhanced control and monitoring applications [6].

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 83–94, 2013.

These new features of automation systems provide solid ground for a new generation of industrial energy management systems, targeting increase in energy awareness through combination of energy consumption and process related data [7]. Moreover, the newly developed models for energy optimization must actively employ the networked and information-reach shop-floor [8]. Internet technologies applied in factories can help the development of holistic understanding of energy performance at manufacturing sites [9]. This is considered to be one of the most important ways ICT could contribute to reduce the energy consumption of manufacturing enterprises [2].

Although the need for holistic perception of plant performance is widely acknowledged, most of the research effort reported concentrates on the design of cross-domain measures [10] and enriched context information [7]. The elaboration of energy saving actions is expected to be implemented by humans (with help of decision support systems). The control strategy aiming the process optimization proposed in [8] considers the impact of different states of the equipment on the energy consumption, yet it does not consider its role in the overall plant performance, because the pursuing of energy optimization should not threaten the production plan.

The real limits of optimization methods are dictated by the operational conditions of the factory, e.g. climate conditions, resource quality, etc. Therefore it is a must that parameters reflecting the operational conditions of the plant are considered in energy management and optimization applications. A Multi-Agent System (MAS) implementation, where each parameter under control is provided with its own Management Agent, is able to handle both the multitude of parameters to be managed and the need to avoid conflicting control instructions.

This paper introduces such a solution, applicable to a practical scenario of a discrete manufacturing plant. The rest of the paper is organized as follows: Section 2 discusses the background of this research, i.e. the technological enablers of the proposed solution, and reviews related works; Section 3 explains the control principles with help of peak-load management scenario using a simulation prototype; Section 4 continues with consideration of requirements and extensions needed for the full-scale implementation; finally Section 5 concludes the paper with summarizing remarks and outline for the future work.

2 Background

In discrete manufacturing, intelligent control relying on the information about site's operational conditions must consider the following aspects:

- The data on the operational conditions is provided by several different systems belonging to the factory automation world, the building automation world and the management world;
- The operating conditions constitute the minimum set of requirements and violations leading to decrease in productivity are not acceptable;
- The parameters describing the operational conditions are interdependent and the control side must consider this interdependence.

As MAS solution is able to provide means for addressing these aspects with help of state-of-the-art ICT tools in the domain, the following subsection introduces the reader to the agent-based control solutions in domain of discrete manufacturing. Then the second subsection is intended to provide background for the control scenario described in section 3 through overview of agent-based applications for energy management.

2.1 Agent-Based Control in Discrete Manufacturing

In the field of discrete manufacturing, agent-based control was initially proposed as a solution to avoid conflicts between the reconfigurability needs of the working environment and productivity levels requirements [11].

But, despite the advantages MAS and Holonic Manufacturing Systems (HMS) offer for addressing the need for flexibility and reconfigurability, they were not fully adopted by the industry, because of a number of known challenges and gaps mentioned by Leitao in [12]: first, the investment required for implementation is higher compared to traditional control strategies; second, there is lack of industrial control devices capable of running the agents along the execution of low level control.

While the limitations mentioned above prevented wide adoption of MAS vision at the device level, the vision was escalated to the MES level of the automation pyramid to facilitate the orchestration of SOA-enabled manufacturing systems. Recent research [13] uses MAS as means of decision support to enhance on-line scheduling through web-service orchestration, with agents representing the atomic web-services exposing equipment capabilities and production orders. Decision support activities (resulting in a newly composed BPM) are triggered whenever an Orchestration Engine encounters the need for dynamic change in Business Process Model (BPM).

From the perspective of the application considered in present paper, the most valuable outcome of the developed agent-based DSS are the recognised extensions of agent platform required for its integration with WS-enabled devices [14]. This enables handling multiple heterogeneous information sources to be involved in control and shift the design focus to detailed consideration of agents' roles and interactions.

But, difficulty of MAS implementations goes beyond simple selection of tools and technologies and system integration, into the need to follow a design methodology facilitating sustainable development [15]. Starting with a number of works proposing consideration of role model for definition of MAS structure (i.e. the agents are designed to fulfil one of the roles identified for a particular case), namely [16-18], Zambonelli et al. propose the considering of organizational rules, structures and patterns [15]. The methodology proposed results in a design specification, which can be implemented using adequate agent-programming framework.

2.2 Agent-Based Solutions for Demand Side Management (DSM) and Power Trading

Power generation peaks have negative environmental impact and decrease overall efficiency of generating plants, as big deviation of generation rates prevents optimization of the process. Therefore peak load avoidance is rather important measure as far as energy efficiency is concerned [2]. This strategy affects both suppliers and the consumers: first can save on construction, operation and maintenance of plants used only to cover power needs during the peak hours and concentrate on optimization of generation process, the latter can benefit from improved reliability and quality of the supplied power. Dynamic DSM is one of the methods helping to flatten the consumption patterns

DSM is a set of measures targeting the improvement of energy systems at consumption side [19]. An approach DSM is demonstrated in the Building-to-Grid (B2M) project [20], in which building automation systems are integrated into a community of cooperating software agents, each representing a building and provided with own role and world model.

Decentralized DSM is proposed in [21]. Agents represent individual smart meters, devices intended to manage household equipment in order to reduce inefficiency and increase savings for the owner. The agents are autonomous and do not communicate with each other.

Li et al. in [22] compare three MAS systems each having three groups of agents: agents representing external devices control the power switch of the appliances; virtual load agents incorporated in smart meters; broker agents, representing intelligent electronic devices in a substation. The agents are organized in three types of structures: decentralized, hierarchical and hybrid, each being evaluated against optimization targets and applicability for real-life applications.

Decentralized structure is called Stigspace and is characterized by stigmeric interactions: *indirect communication mediated by modifications of the environment* [23]. While study considering the stigspace structure confirmed its applicability for DSM [24], the comparison of different structures suggested selection of hybrid approach [22], which leads to the conclusion, that at least minimum coordination among agents is required.

Earlier and more traditional applications of agent-based technology in the energy domain are concerned with electricity trading. Agent-based solutions are actively used to evaluate pricing strategies and power trading policies. Xiong et al. in [25] discuss uniform pricing and pay-as-bid auctions with the help of adaptive agents representing bid price generators using Q-Learning algorithm. The agents learn from their own bidding experience and aiming at maximizing their profit in long term. Bunn and Oliveira in [26] apply agent-based simulation to evaluate new electricity trading arrangements. In this solution agents represent system operators, generation companies, and suppliers (mediators between the wholesale market and end users), they are autonomous agents and do not have communication capabilities.

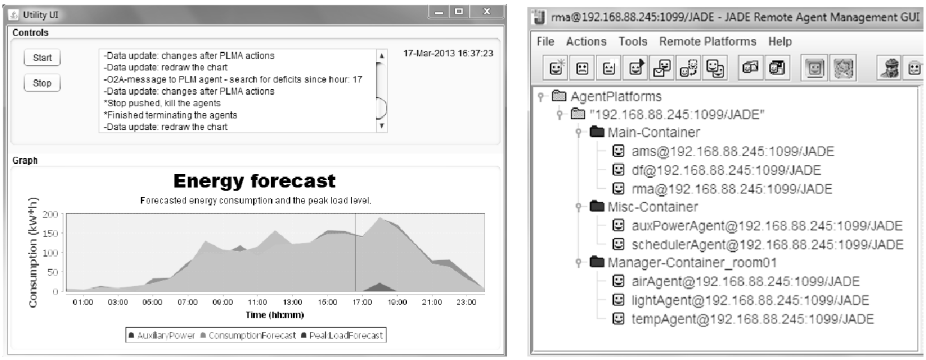
Almost all of the studied examples suggest use of autonomous agents, imitating the actual organizational patterns in the domain. However, studies prove the need for at least minimum communication among agents to make solution applicable for real-life deployment.

3 Control Scenario: Peak Load Management

The objective of the discussed intelligent control system is the resource consumption optimization of the manufacturing site without compromising operational conditions.

To meet operational conditions, a number of parameters (henceforth referred to as Site Dimensions (SD)) must be maintained within specified limits. The limits are defined by the operational conditions of the process and can be directly specified by the equipment settings. SD values depend on the amount of resources required by the technical systems fulfilling the dimension. The resource here is understood as matter, substance, phenomenon, being, material involved in site operation, yet not the equipment or structures. Change in the amount of resources results in change of values of the site dimensions. The nature of the problem suggests applying MAS for control implementation.

Realizing importance of collaboration between agents in the considered control application, a small-scale agent-based simulation was implemented to study interaction patterns and identify common categories of agents. The objective was to implement a scenario illustrating the control principle on a dimension clearly affecting the energy performance of the plant and illustrate message flows concerned with decision-making. Simulation illustrates control scenario using the peak load management example. The *peak load* dimension is constrained by the capacity of the utility and is crucial for the manufacturing process, as it ensures there is enough power for the machines to run.



(a) (b)

Fig. 1. Screenshot of running simulation: a – Utility; b – JADE RMA GUI listing agents involved in simulation

As opposed to agent-based solutions for DSM based on the autonomous agents discussed in the background section, the control scenario presented in this section relies heavily on collaborating agents. The prototype platform was implemented using JADE (Java Agent Development) Framework [28] and a simple desktop application (Fig. 1.a) was developed to visualize the outcomes of the control using Java programming language.

The Graphical User Interface (GUI) of the Utility consists of two zones: controls and graph. Controls allow user to start and stop the simulation and follow the status messages in the log. Graph displays the amount of the electrical power available from the utility for the following 24 hours and forecasted consumption rates according to the production schedule. This information is supplied to the Peak Load Management Agent (PLMA).

Evaluation of the input data by PLMA results in three possible reactions:

- no action is required if there is no power deficit detected;
- minor deficit detected and the *minor deficit behaviour* is triggered;
- big deficit detected and the *noticeable deficit behaviour* is triggered.

3.1 Minor Deficit Behaviour

In case of the minor deficit PLMA sends the duration and deficit amount to the collaborating energy manager agents (i.e. *CompressedAirAgent*, *TemperatureAgent* and *LightingAgent* of *Manager-Container_room01* in Fig. 1.b). This will trigger a series of actions, as illustrated in sequence diagram on Fig. 2.a. First, as manager agent receives the *reduction-request* it evaluates the data against its own current process set point and its limits for the current process, and calculates the lowered set point that would reduce the energy consumption of the process and cover the deficit, or part of it. The manager agent searches for the other manager agents who have registered their service to the Directory Facilitator (DF) of JADE and sends the proposal of the new set point to them. The conversation-id is *reduction-proposal* and message format is *"'the amount of hours ahead from this moment'; 'the new set point'"*. If initially no reduction is possible, the manager agent sends a *refuse* message to the PLMA.

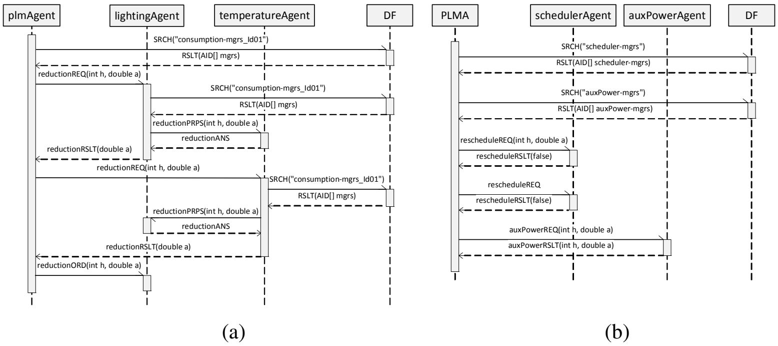


Fig. 2. Handling different types of power deficit: a – minor deficit behaviour; b – noticeable deficit behaviour

Each time a manager agent receives a proposal from another manager agent it has the right to either agree or refuse the proposal. If there is an agent that refuses the proposal, the initial sender of the proposal cannot carry out this set point change, because the change negatively affects process managed by the refusing agent. If all of the answers are positive, the energy manager agent sends the amount of possible energy consumption reduction to the PLMA. The conversation-id of this message is *reduction-result*, and the message consists of the amount of possible reduction.

A specific class was implemented to handle the inter-process set point limitations in this simulation. The *TransferFunctionTable* class stores a map structure from which the set point limitations are searched by using the agent ID of the proposing agent and the agent ID of the agent who received the proposal.

The PLMA does not send the proposals to all of the energy manager agents simultaneously, even though this would be possible. It was noticed that if agents receive multiple messages of the same conversation-id, the message received later overwrites the one received earlier. Therefore PLMA waits for an answer before sending a new proposal.

When the PLMA has received an answer from each of the energy manager agents it sent the reduction-proposal to, it processes the possible reduction data and decides which agents should change their set points, and how much they need to release energy consumption. This message has *reduction-order* conversation-id and the message is of format *"'the amount of hours ahead from this moment'; 'the amount of energy consumption to release'"*. The energy manager agents cannot refuse to carry out this request.

3.2 Noticeable Deficit Behaviour

In the second case the amount of deficit exceeds the acceptable threshold. In this situation the deficit can be handled either by rescheduling the production or by involving auxiliary power sources, as illustrated in Fig. 2.b.

The PLMA will always first request the *SchedulerAgent* for rescheduling the production. The message's conversation-id is *reschedule-request* and the message is of form *"'the amount of hours ahead from this moment'; 'the amount of energy consumption to release'"*. After receiving the message the *SchedulerAgent* should start its rescheduling function that determines how rescheduling could affect the energy consumption at this moment and in future.

The PLMA requests for rescheduling twice and if at the both times rescheduling fails, the involvement of auxiliary power source is requested. *AuxPowerAgent* is responsible for utilizing the auxiliary power sources. It always carries out the requests it receives.

4 Implications for Plant-Scale Agent-Based Control

In order to achieve resource consumption optimization of the manufacturing site without compromising operational conditions, the following requirements must be considered:

(1) the control strategy must accept multiple input variables and synchronize elaborated commands with the objectives of the affected systems;

(2) a common approach to representation of heterogeneous information is needed in order to provide input data in a uniform manner;

(3) the information capturing methods must consider the variety of sources to be included in order to obtain all necessary data.

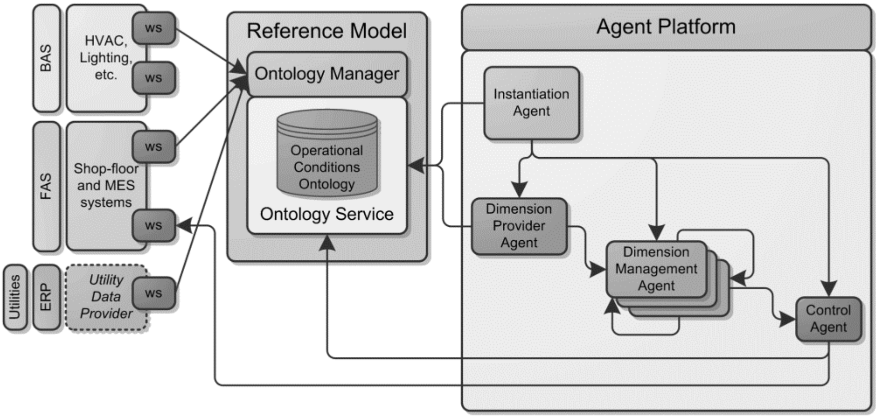


Fig. 3. Components of agent-based control system

Following these requirements, three classes of components can be identified: *Information sources* providing input data, *Reference model* representing information about the site in format aligned with the specifics of the control application and the *Agent platform*, containing agents taking the control decisions. The controlled systems physically coincide with the input data providers, except the utilities, which are outside of the scope of the control application.

4.1 Information Sources

Multiple input variables are fed into the control system from three major sources of information:

- *Factory Automation Systems (FAS).* The information sources of FAS are shop-floor devices and Manufacturing Execution Systems (MES) applications. Shop-floor devices (smart meters and PLCs) provide mainly process-related measured values and Key Performance Indicators (KPI) limited to the device scope. The applications at MES level may offer site-scoped KPIs and data aggregated from field devices.
- *Building Automation Systems (BAS).* Similar to the case of FAS, the information required from the BAS comprises the data about ambient conditions of the site (measured) and operating modes of HVAC and lighting systems. Additionally aggregated KPIs may be requested from the monitoring applications.
- *Utility information.* The information exchange between utilities and the manufacturing sites happens mainly at Enterprise Resource Planning (ERP) level and is concerned with resource procurement and service provisioning. As operational conditions are assured with help of resources provided by the utilities, information concerning resource availability and quality is needed for control.

Despite the heterogeneity of the sources described above, the information can be accessed in uniform manner. With the adoption of SOA and SWS by the industry, the access to the information from both shop-floor and MES is possible via web-service interfaces. Similarly, most information concerning the environment of the manufacturing site can be accessed directly from WS-enabled controllers. The biggest advantage of the approach is that information can be accessed in an event-driven manner, reducing sufficiently the network load and unnecessary communication.

ERP information systems are not heavily constrained with equipment performance limitations and a secure and reliable interface for the application discussed can be implemented. The access to the utility supplied data may be provided in form of a web service interface either to the application itself, or to the intermediate information storage, in order to keep the data provider decoupled from the consumer.

4.2 Reference Model

The reference model is intended to organize the information regarding the manufacturing site in a form, allowing the tracking of KPIs of interest with respect to the overall site structure and its components. Ontologies are very suitable to represent such reference models, having capability to represent only existing features and dependencies, yet keeping a reasonable level of generality. They also offer procedures defined for the dynamic updates of the model at the individual instance level. Moreover, ontologies have been widely used in the implementation of SWS applications, considered a technological prerequisite for integration of the information sources.

The ontology is proposed to model the factory environment in a holistic manner from the operational conditions point of view. At taxonomy level it defines hierarchies of technical systems, site dimensions, resources and defines basic relations between them. The individuals represent specific KPIs, resources, systems. For dedicated technical systems the proposed ontology contains definitions of the web-service operations to be invoked in order to execute the control command.

The ontology is contained within the ontology service, the component intended to provide interface for remote querying and update of the model. The model is kept updated with recent information from the environment with help of the Ontology Manager tool, which evaluates received messages against the set of update rules [27].

4.3 Agent Platform

In contrast to the MAS applications for control of reconfigurable manufacturing systems, all the agents can be deployed in one platform, which simplifies significantly the agent design. The agents within the platform can communicate with the reference model via web-service interface of the ontology service [13].

There are four types of agents in addition to the platform's default agents: (1) Instantiation Agent, (2) Dimension Provider Agent (DPA), (3) Dimension Management Agent (DMA), and (4) Control agent (CA).

The Instantiation Agent is created upon the platform start. It queries the reference model in order to identify types and amounts of management agents to be deployed for covering the site. Then it creates the DMAs, a DPA and a CA.

DPA is intended to monitor the status of the system for changes, by evaluating the changes in the ontology. After retrieval of the information it evaluates the values and plans the messaging following the established priority. First message is being sent to the DMA in charge of the most important dimension. The next message will be sent only if no potential changes are reported for the other dimensions by the contacted DMA. Otherwise, if system status changes after the control has been applied, the message queue is cleared.

A management agent is created for each dimension requiring intelligent control. Each agent is assigned a priority in order to avoid conflicting control strategies. There are two main behaviours defined for each DMA: dimension management behaviour and control assessment.

Dimension management. The behaviour consists of several stages. First agent receives a message from DPA with updated dimension value, and then it evaluates the value with respect to the criteria, specified for the dimension under its responsibility and decides on action to be taken. The action planned is advertised to the other management agents, triggering the control assessment behaviour. If all the agents approve the action, the command is sent to the control agent.

Control assessment. Agent receives the advertised control command and evaluates its impact on the dimension under its responsibility and either accepts or rejects the proposal.

The control agent receives the elaborated command from DMAs and queries the model for the details of operation invocation, i.e. operation name and parameters. Then it invokes the operation on the dedicated system in order to execute the command.

Following this approach platform has two synchronization points: at the input in DPA and at the output in Control Agent. This allows avoiding conflicting control commands and reduces significantly the amount of messages to be sent.

5 Conclusion

This paper discusses agent-based control for the management of peak load scenarios to exemplify and motivate control of operational conditions as means to obtain holistic understanding of site performance. In particular, load management is inspected under minor and noticeable deficit cases, considering different set of actors involved in decision making in each of the case.

The proposed approach provides a holistic view on operational status of the manufacturing site; moreover it contains implicit information about the site's energy footprint.

Future work will investigate visualization tools reflecting adequately the information embodied in the model and associated control decisions and integration with utility information systems (e.g. for energy systems and smart grids).

References

1. World Bank Data. Manufacturing, value added (% of GDP). Link: http://data.worldbank.org/indicator/NV.IND.MANF.ZS/countries?display=graph (accessed on March 02, 2013)
2. ICT and Energy Efficiency. The Case for Manufacturing. Recommendations of the Consultation Group. European Commission (February 2009) ISBN 978-92-79-11306-2
3. Energy Use in the New Millennium. Trends in IEA Countries. International Energy Agency (IEA) (2007)
4. Martinez Lastra, J.L., Delamer, I.M.: Semantic web services in factory automation: fundamental insights and research roadmap. IEEE Trans. Industrial Informatics, 1–11 (2006)
5. Cannata, A., Gerosa, M., Taisch, M.: SOCRADES: A framework for developing intelligent systems in manufacturing. In: Transactions of IEEE International Conference on Industrial Engineering and Engineering Management, IEEM 2008, pp. 1904–1908 (2008)
6. Postelnicu, C., Zhang, B., Martinez Lastra, J.L.: Embedded Service Oriented Monitoring for the Energy Aware Factory. In: 2012 6th International Conference on Information and Automation for Sustainability (2012)
7. Heilala, J., Klobut, K., Salonen, T., Siltanen, P., Ruusu, R., Armijo, A., Sorli, M., Urosevic, L., Reimer, P., Fatur, T., Gantar, Z., Jung, A.: Ambient Intelligence based monitoring and energy efficiency optimization system. In: 2011 IEEE International Symposium on Assembly and Manufacturing (ISAM), May 25-27, pp. 1–6 (2011)
8. Cannata, A., Karnouskos, S., Taisch, M.: Energy efficiency driven process analysis and optimization in discrete manufacturing. In: 35th Annual Conference of IEEE on Industrial Electronics, IECON 2009, November 3-5, pp. 4449–4454 (2009)
9. Karnouskos, S., Colombo, A.W., Martinez Lastra, J.L., Popescu, C.: Towards the energy efficient future factory. In: 7th IEEE International Conference on Industrial Informatics, INDIN 2009, June 23-26, pp. 367–371 (2009)
10. Herrmann, C., Thiede, S.: Process chain simulation to foster energy efficiency in manufacturing. CIRP Journal of Manufacturing Science and Technology 1(4), 221–229 (2009)
11. Martinez Lastra, J.L., Colombo, A.W.: Engineering framework for agent-based manufacturing control. Engineering Applications of Artificial Intelligence 19(6), 625–640 (2006)
12. Leitão, P.: Agent-based distributed manufacturing control: A state-of-the-art survey. Engineering Applications of Artificial Intelligence 22(7), 979–991 (2009)
13. Villaseñor Herrera, V., Vidales Ramos, A., Martínez Lastra, J.L.: An agent-based system for orchestration support of web service-enabled devices in discrete manufacturing systems. Journal of Intelligent Manufacturing 23(6), 2681–2702 (2012)
14. Villaseñor Herrera, V., Bepperling, A., Lobov, A., Smit, H., Colombo, A.W., Martinez Lastra, J.L.: Integration of Multi-Agent Systems and Service-Oriented Architecture for industrial automation. In: 6th IEEE International Conference on Industrial Informatics, INDIN 2008, July 13-16, pp. 768–773 (2008)
15. Zambonelli, F., Jennings, N.R., Wooldridge, M.: Organisational Rules as an Abstraction for the Analysis and Design of Multi-Agent Systems. International Journal of Software Engineering and Knowledge Engineering 11(3), 303–328 (2001)
16. Demazeau, Y., Rocha Costa, A.C.: Populations and organizations in open multi-agent systems. In: 1st National Symposium on Parallel and Distributed AI, PDAI 1996 (1996)
17. Ferber, J., Gutknecht, O.: A meta-model for the analysis and design of organizations in multi-agent systems. In: Proc. 3rd Int. Conf. on Multi-Agent Systems (ICMAS 1998). IEEE CS Press (June 1998)

18. Kendall, E.A.: Role modelling for agent system analysis, design, and implementation. In: 1st Int. Symp. on Agent Systems and Applications. IEEE CS Press (October 1999)
19. Palensky, P., Dietrich, D.: Demand Side Management: Demand Response, Intelligent Energy Systems, and Smart Loads. IEEE Transactions on Industrial Informatics 7(3), 381–388 (2011)
20. B2G - Smart Grids Model Region Salzburg – Building to Grid. Link: `http://energyit.ict.tuwien.ac.at/index.php/en/projects/b2g` (accessed on February 20, 2013)
21. Sarvapali, R., Perukrishnen, V., Rogers, A., Jennings, N.: Agent-Based Control for Decentralised Demand Side Management in the Smart Grid. In: The 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 02-06, pp. 5–12 (2011)
22. Li, R., Li, J., Poulton, G.T., James, G.C.: Agent based optimisation systems for electrical load management. In: Proceedings of 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008): 1st International Workshop on Optimisation in Multiagent Systems (OptMAS 2008), Estoril, Portugal, pp. 60–69 (2008)
23. Marsh, L., Onof, C.: Stigmergic epistemology, stigmergic cognition. Cognitive Systems Research 9(1-2), 136–149 (2008)
24. Li, J., Poulton, G., James, G.: Agent-Based Distributed Energy Management. In: Orgun, M.A., Thornton, J. (eds.) AI 2007. LNCS (LNAI), vol. 4830, pp. 569–578. Springer, Heidelberg (2007)
25. Xiong, G., Okuma, S., Fujita, H.: Multi-agent based experiments on uniform price and pay-as-bid electricity auction markets. In: Proceedings of the 2004 IEEE International Conference on Electric Utility Deregulation, Restructuring and Power Technologies, DRPT 2004, vol. 1, pp. 72–76 (2004)
26. Bunn, D.W., Oliveira, F.S.: Agent-based simulation-an application to the new electricity trading arrangements of England and Wales. IEEE Transactions on Evolutionary Computation 5(5), 493–503 (2001)
27. Puttonen, J., Lobov, A., Cavia Soto, M.A., Martinez Lastra, J.L.: A Semantic Web Services-based approach for production systems control. Advanced Engineering Informatics 24(3), 285–299 (2010)
28. Jade - Java Agent DEvelopment Framework. Link: `http://jade.tilab.com/` (accessed on February 20, 2013)

IceHMS, a Middleware for Distributed Control of Manufacturing Systems

Olivier Roulet-Dubonnet[1], Morten Lind[1,2], and Amund Skavhaug[3]

[1] Department of Production Technology, SINTEF Raufoss Manufacturing
olivier.roulet-dubonnet@sintef.no
[2] Department of Production and Quality Engineering,
The Norwegian University of Science and Technology
morten.lind@sintef.no
[3] Department of Applied Cybernetics, The Norwegian University of Science and Technology
Amund.Skavhaug@itk.ntnu.no

Abstract. This paper presents a new middleware, dubbed IceHMS specifically designed to support the development of distributed manufacturing control systems requiring flexibility. IceHMS has been used in several research applications in the field of flexible and reconfigurable manufacturing which are documented in dedicated scientific papers. This paper focuses on the description of the IceHMS middelware.

IceHMS distinguishes itself from mainstream multi-agent platforms by its design for distributed applications and in its cross-programming language and cross-OS (operating system) support. IceHMS is a thin, non-intrusive multi-agent layer running over a proven communication engine, the Internet Communication Engine (Ice™) by ZeroC Inc. The code of IceHMS is open-source and is freely available.

Keywords: Multi-Agent System, Holonic Manufacturing System, multi-agent platform.

1 Introduction

In order to deal with the complexity arising from desired flexibility and reconfigurability of manufacturing systems, one strategy regards distributed autonomy in the control system. Physical distribution of computational resources is a necessity to extend computing power and fulfil real-time constraints. Distribution and autonomy are therefore accepted by many researchers as a necessity to develop flexible and reconfigurable manufacturing systems; confer e.g. Scholz-Reiter and Freitag [23].

The available software technologies for the development of distributed systems for manufacturing control can be roughly categorized into the following types: lower level communication middleware, specialized multi-agent or holonic [28] platforms, and robotic platforms.

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 95–105, 2013.

1.1 Related Work

There exist several recent review papers on Multi-Agents and Holonic Manufacturing System. Few of these focus on technologies but most mention them. A survey of specialized programming languages and platforms for Multi-Agent Systems is given by Bordini et al. [4]. The review presents technologies for implementing Multi-Agent Systems with focus on logic and formal methods. Monostori et al. [17] give an in-depth review of agent-based systems for manufacturing and present many domain concepts such as Multi-Agent System, ontology, Holonic Manufacturing System and the different control levels in manufacturing. Another comprehensive review is presented by Shen et al. [24], who dedicate a section to surveying multi-agent tools and languages. Leitão [13] reviews applications of multi-agent and holonic systems for manufacturing control. He makes frequent mention of the technologies used for the presented applications. A survey of Multi-Agent Systems technologies is also to be found in the paper by Vallejo et al. [27].

JADE [2] seems to be the most used multi-agent platform in manufacturing research [1, 13]. The platforms benefits from a dynamic community of contributors and users, with many specialized versions available. JADE is a multi-agent platform implemented in Java and thus, in principle, executable on every OS and hardware where a Java run-time machine is available. Like JADE, Cougaar [8] and JACK [5] are also Java based multi-agent platforms. In fact, most of the recent multi-agent platforms are Java based [24].

A conceptually different, though architecturally similar, set of middleware technologies relate to robotics. These technologies are developed to control a robot system or installation with its associated sensors and peripheral devices. Modern robotic systems may be highly complex, comprising many devices, sensors and processing units linked together with an internal networked system. They are therefore similar to shop-floor systems with respect to some aspects, and thus relevant to review in this paper. A modern example of robotic middleware is ORCA [16]. ORCA is a component-based middleware, originally based on the Common Object Request Broker Architecture (CORBA) but now using Ice™ [10].

The Robot Operating System (ROS) [18] is a another example of a modern middleware to control an advanced robot. It is highly popular in academic research laboratories. ROS has a centralized localization service but the communication is peer-to-peer once established. It supports three basic communication mechanisms: (service) procedure invocation, messages, and publish/subscribe. It supports several programming languages to satisfy individual developer preferences, and focuses on staying thin and non-invasive in order to encourage the development of middleware agnostic libraries. It is tool based, i.e. it does not offer a central *Integrated Development Environment* (IDE) but many small tools to perform their dedicated tasks.

Due to their focus, ROS and many other robotic frameworks are affected by design choices that reduce their scalability and make them cumbersome to use for shop-floor control. ROS, for example, offers no object oriented interfaces for method invocation on nodes, but only plain service procedure calls. Hence, the set of service procedures offered by each node must be differentiated by name from the same service procedures on the other concurrent nodes.

An alternative approach to using a specialized platforms is to use a lower level communication middleware. To the authors' knowledge, this is often the chosen solution in industrial systems. CORBA is, perhaps, the best known cross-platform, cross-language communication standard for location-transparent method invocation. Nevertheless, CORBA seems to have lost some of its glory and more recent alternatives are available; e.g. ZeroC Ice™ [10]. If the target application or platform does not require cross-platform or cross-language feature, many alternative communication technologies become relevant; e.g. Java RMI, *.NET Remoting*, or Python Pyro [3].

Erlang/OTP [12] is another low-level technology which is sometime mentioned to develop industrial Multi-Agent System, confer for example to Varela et al. [29]. Erlang is a functional programming language focussing on concurrent, distributed systems, and tightly coupled to its application platform called Open Telecom Platform (OTP). Erlang was developed in the 1980s and has remained obscured for most of its existence, but is now gaining some renaissance due to the current focus on concurrency. To achieve its goal, Erlang was designed with characteristics such as message-passing concurrency, single assignment and functional model, which may restrict its use as a general-purpose multi-agent programming language.

There are many prototype multi-agent platforms in the research literature, ranging from high-level systems with automatic code generation to real-time OS kernel-level based systems. A relevant work is the Jabber-based multi-agent platform presented by Gregori et al. [7]. It is developed on top of the *Extensible Messaging and Presence Protocol* (XMPP), formerly named Jabber, which is a communication protocol developed for online messaging over the Internet. The platform has the advantage of being distributed, decentralized, and with client libraries implemented for most OSs and in most programming languages; and support cross-OS and cross-programming language Multi-Agent Systems. The platform includes a plugin to inter-operate with JADE.

Vallejo et al. [27] developed a FIPA-compliant [6] multi-agent platform. The multi-agent platform is implemented using the Ice™ middleware [9] and is distributed as open and free software. Ice™ itself is released under the GNU General Public License (GPL), and is a modern communication middleware with cross-support for many programming languages and OSs. According to its creators, Ice™ is used in many mission-critical projects by companies all over the world. Ice™ has a rich feature-set, offering persistence, replication, location service, a publish/subscribe mechanism, etc. As a result, the multi-agent platform implementation remains relatively thin.

1.2 Motivation for a New Middleware

Today's shop floors are a collection of heterogeneous systems. They contain hardware devices ranging from special machines, to many different industrial robots and CNC machines. Shop floors also contain a flora of software products from wide variety of manufacturers. Communication among these systems ranges from non-existent to very simple.

A key task of distributed manufacturing control systems is to control and therefore integrate a potentially large number of heterogeneous hardware and software entities. However the mentioned solutions are not adapted to this task:

- First the control system must be physically distributed between many computing units. To the authors' knowledge, while all the multi-agent platforms mentioned in the last section support the development of Situated Multi-Agent Systems [30], they consider single agents located on their own physical unit as a corner case. Their scalability and usability is therefore often inadequate in this situation.
- Cross-OS, cross-programming language support, and absence of strong application platform obligations are highly advantageous properties when integrating a heterogeneous collection of hardware devices and software systems. Such properties in the integrating platform allow for the direct integration of entities coming with their native control systems or native libraries, rather than being forced to developing custom protocol or thick wrappers, adaptors, mediators, etc.
- An important aspect of shop-floor control is that most parts of the control systems have timeliness constraints. The chosen technologies underlying the control system must therefore have performance characteristics which do not prohibit the fulfilment of such constraints. TCP/IP based systems do not offer hard real-time characteristics. However by knowing its limitations, giving the programmer the necessary flexibility to tune the communication between holons and carefully designing the applications, it is possible to use TCP/IP for shop-floor control; confer e.g. [25].

A promising solution is the Ice™ based, FIPA compliant, platform from Vallejo et al.. Standardization is highly desirable when implementing controllers on industrial sites. However, the integration of FIPA at the core of the multi-agent platform, imposes several architectural constraints on the distributed application. This motivated the design and implementation of a new middleware, inspired by the work of Vallejo et al. but giving the application developers more flexibility; by allowing direct use of features of Ice™ from the logic code level, such as method call and transport layer set-up. The middleware has been called IceHMS in reference to the underlying technology (Ice™) and the inspiring paradigm (Holonic Manufacturing System).

During the last years, IceHMS has been been used in several applications. The most relevant are documented in the following research papers: An application to control an automated paint shop[14], a flexible simulation system based on blender[15], an automated assembly cell for damped machining tools[22], an hybrid manufacturing cell combining subtractive and additive machines[19], and an application to use a commercial discrete event simulation platform to simulate and validate distributed control systems [21]. In this paper the motivation, features and functioning of IceHMS are documented.

2 Overview of IceHMS

IceHMS is middleware to support the development of distributed holonic or multi-agent systems. The code of IceHMS is open-source and published under the GNU GPL licence [26]. It can be freely downloaded, modified and used [20].

The following properties characterize the newly developed multi-agent middleware:

– Its API and architecture is designed for distributed applications, with one holon per computing unit as a common case.
– It enables holons to find each others and communicate either peer to peer or through publish subscribe servers. It support both message based communication and method call.
– It includes a set of features, tools, and APIs for creating, deploying, and managing the distributed manufacturing control entities.
– Its performance scales well, from control systems at device or cell level to distributed systems for complete enterprise production control.
– It is cross-programming language and cross-OS supportive, and thus isolates the choice of language and OS at the individual agent or entity level.
– It contains no tight application-platform, which might otherwise impose constraints in the technology selection, design, implementation, or deployment phases.

An overview of the architecture can be seen in Figure 1. The middleware is composed of two parts: 1) the core system, containing the necessary infrastructure and interface definitions, confer Section 3, and 2) the language specific utility classes to interact with Ice$^{\text{TM}}$, confer Section 4.

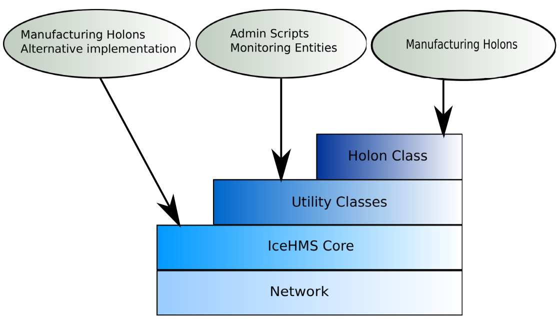


Fig. 1. High level overview of IceHMS

In order to support the development of large distributed control systems, special care has been taken to make IceHMS scalable. The communication between the nodes is direct, there is not central message routing. Method calls between holons are location independent, Ice™ transparently chooses the most adapted transport between network, local socket and direct call. IceHMS also allows for scalability in the number of agents that can be run on one processing unit. By default, when a large number of agents are created under the same AgentManager (see Section 4), they use a common Ice™ adapter for communication. Thus, method calls between these agents are *short-cut inside the Ice run time and dispatched directly* [31]. When the agents are spread over the network, the main potential congestion point, apart from the network design and computing power available, is the name service. However, the name service is based on a scalable and proven technology; for details, see Section 3.3 and Henning [11].

3 The Core System

The core of IceHMS is composed of the 'yellow page'/service discovery, name service, administrative programs and central publish/subscribe servers. It also contains default interfaces for the holon objects and other basic types, as well as a default Ice™ configuration for communication. The core system of IceHMS can be used directly in all programming languages and from all OSs supported by Ice™.

3.1 Interfaces Declaration

IceHMS comes with a set of default interfaces to support message based communication and the transfer of some common objects. Projects that are not using the default message interface can declare their own objects and interfaces inheriting the Holon interface. The interfaces can be declared using the *slice* language as defined in the Ice™ middleware. Interfaces are language independent and the objects and methods can be implemented using any Ice™-supported programming language. Listing 1.1 shows the interface definition of the holon object, which is by default minimal, and can be extended at will by the applications:

```
interface MessageInterface {
    ["ami"] void put_message(Message s);
};

interface Holon extends MessageInterface{
    string get_name();
};
```

Listing 1.1. The default interface definition of a Holon

3.2 Publish/Subscribe Servers

IceHMS comes with a set of system-wide publish/subscribe servers. It has a default server with a message interface which allow holons to communicate easily in a centralized way as implemented in many multi-agent systems. IceHMS has also a second server which has a configuration adapted to communication in real-time such as data streaming from a sensor. For example, the communication with this server uses by default UDP instead of TCP. Finally it is possible to set-up additional servers for local communication.

3.3 Name Service

A critical functionality of a middleware for distributed control is to allow holons to find each other. This is offered by the name service and the service discovery features.

At a lower level finding a holon mean knowing its address. To obtain holon addresses, IceHMS uses a centralized but proven and scalable solution based on the default name services of Ice™ called IceGrid. Since the name of the holon is used as the key identifier, it must be unique.

As with internet name services, the location of the root name service, its IP address and port, must be given to any application that needs to interact with the manufacturing system. In the Python wrapper this is done using an environment variable or programmatically.

3.4 Service Discovery

The role of a service discovery system is to locate other holons by knowing their services or properties. IceHMS implements a simple solution where holons can be found knowing their inherited types. In the current implementation, a simple string, the default Ice™ type property, is used as the type of the holon and searches are executed using the default query interface of IceGrid. This solution has the advantage of relying on proven code and results in a scalable system. However, it is limited since it only allows to search for holons by their inherited type and does not allow for the definition of custom attributes.

4 The Utility Classes

The utility classes offer a higher level interface to IceHMS. They simplify the initialization and the routine operations. Still, in order to give programmers flexibility, the lower level objects are exposed and the Ice™ middleware can be used directly by the holons. A class diagram of the classes can be seen in Figure 2 and Figure 3 documents the startup sequence. It shows interaction of the main classes in IceHMS. These classes are currently only implemented for the Python and C# programming languages.

4.1 The IceManager Class

The IceManager sets up the Multi-Agent System. Most of the logic in the IceHMS utility classes resides in the IceManager class. To increase flexibility the IceManager sets programmatically the necessary lower level Ice™ properties which are per default in text files in an Ice™ application.

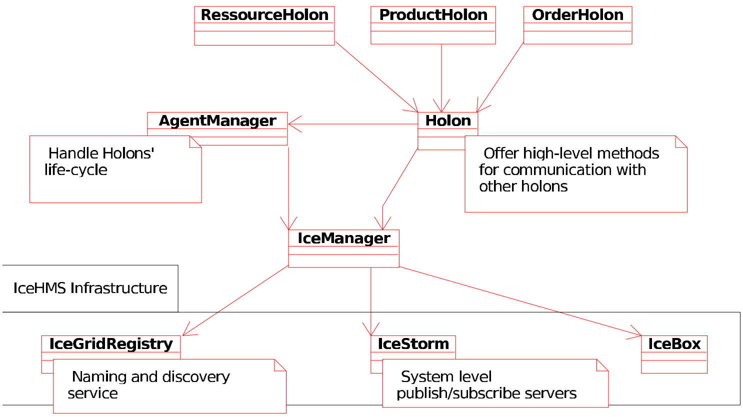


Fig. 2. Simplified UML class diagram of the main Python utility classes. The holon classes at the upper right are the fundamental classes of PROSA.

The IceManager also contains several methods to interact with the Multi-Agent System such as to manage topics (publish/subscribe), search for holons, and make proxies for holons found in searches. It also keeps pointer to Ice™ objects, thus programmers with knowledge of Ice™ can interact directly with the Ice™ middleware.

4.2 The AgentManager Class

The AgentManager object handles the holons' life cycles. It ensures that they are started and shutdown cleanly, and registers holons for name service and service discovery. It creates an IceManager object to communicate with the Multi-Agent System and catches signals from the OS.

4.3 The Holon and LightHolon Classes

The Holon class is the base class. It is expected that holons classes derives from the Holon class. When inheritance is not desired, it is possible to directly use an instance of IceManager to communicate with the Multi-Agent System. The difference between Holon and LightHolon is that the Holon class derives from the Python Thread class, giving it a thread of execution for life cycle tasks.

The code example in Lst. 1.2 is a complete Python implementation of a holon using IceHMS. It can be run directly from the Python interpreter and does not require any platform, as opposed to Jade for example. The code also illustrates the search for holons on the network, the subscription to published messages from another holon and the direct call of a method from another holon: *get_name()*. It is also to be noted that the *wait_for_shutdown()* method is optional, IceHMS does not need the main process thread, it is available to the programmers and other libraries.

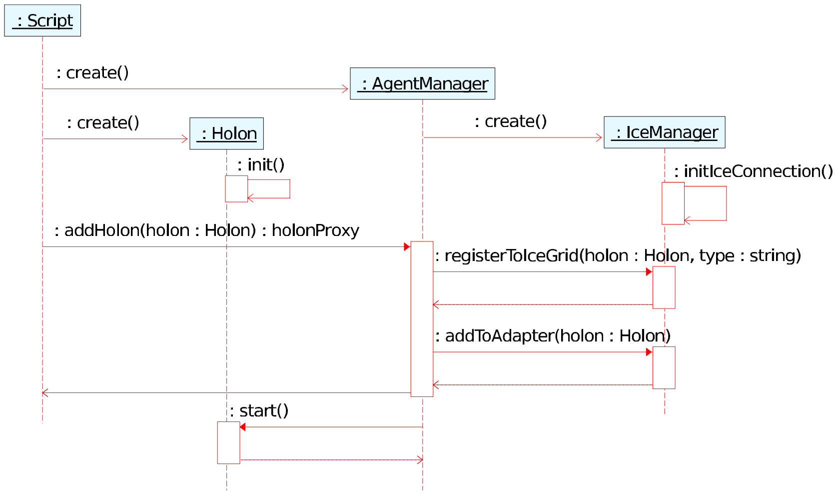


Fig. 3. Simplified startup sequence for a Python script

```
from icehms import Holon, AgentManager

class MyHolon(Holon):
    def run(self):
        robots = self._icemgr.find_holons("hms::Robot")
        self._subscribe_topic("MyScheduler")
        for robot in robots:
            print("I found a robot called: ", robot.get_name())
        if len(self.mailbox) > 0:
            print("I got a message", self.mailbox.pop())

mgr = AgentManager()
holon = MyHolon("MyDevice")
mgr.add_holon(holon)
mgr.wait_for_shutdown()
```

Listing 1.2. The implementation of a standalone holon in Python using IceHMS

5 Conclusion

This paper has presented IceHMS, a new middleware for flexible and reconfigurable manufacturing control systems. IceHMS has been used in several applications and has appeared to be flexible and reliable. IceHMS is especially adapted to the development of control systems which require physical distribution, autonomy, and integration with external systems, typically integrating the control of a large number of heterogeneous devices. However, it is also able to support more classical scenarios where many agents are deployed in one computing unit in order to solve a specific computational problem, such as production scheduling.

It remains an effort of future work to make controlled experiments for inferring the detailed performance in term of scalability, reliability, and latency. Though free messaging is already implemented for holons in IceHMS, a more formal support for the FIPA standard will most likely be added in the near future.

In addition, the utility classes will be implemented, and made available, for more of the programming languages that are supported by Ice™.

Acknowledgements. This paper presents research results obtained through work in the IntelliFeed project and SFI Norman program, both funded by the Research Council of Norway.

References

1. Bellifemine, F., Caire, G., Poggi, A., Rimassa, G.: JADE: A software framework for developing multi-agent applications. Lessons learned. Information and Software Technology 50(1-2), 10–21 (2008)
2. Bellifemine, F., Poggi, A., Rimassa, R.G.: JADE - a FIPA-compliant agent framework. In: Proceedings of the Practical Applications of Intelligent Agents. The Practical Application Company Ltd. (1999)
3. Blank, D., Kumar, D., Meeden, L., Yanco, H.: Pyro: A python-based versatile programming environment for teaching robotics. Journal on Educational Resources in Computing (JERIC) 3(4), 1 (2003)
4. Bordini, R.H., Braubach, L., Dastani, M., Fallah-Seghrouchni, A.E., Gómez-Sanz, J.J., Leite, J., O'Hare, G.M.P., Pokahr, A., Ricci, A.: A survey of programming languages and platforms for multi-agent systems. Informatica, 33–44 (2006)
5. Busetta, P., Ronnquist, R., Hodgson, A., Lucas, A.: Jack intelligent agents-components for intelligent agents in java. AgentLink News Letter 2, 2–5 (1999)
6. Foundation for Intelligent Physical Agents: Fipa abstract architecture specification (2002), http://www.fipa.org/specs/fipa00001
7. Gregori, M.E., Cámara, J.P., Bada, G.A.: A jabber-based multi-agent system platform. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2006, pp. 1282–1284. ACM, New York (2006), http://doi.acm.org/10.1145/1160633.1160866
8. Helsinger, A., Wright, T.: Cougaar: A robust configurable multi agent platform. In: 2005 IEEE Aerospace Conference, pp. 1–10. IEEE (March 2005)
9. Henning, M.: A new approach to object-oriented middleware. IEEE Internet Computing 8(1), 66–75 (2004)
10. Henning, M.: The rise and fall of CORBA. Queue 4(5), 34 (2006)
11. Henning, M.: Choosing Middleware: Why Performance and Scalability do (and do not) Matter. zeroc Inc. (2009)
12. Armstrong, J., Virding, R., Williams, M., et al.: Erlang programming language (1986), http://www.erlang.org
13. Leitão, P.: Agent-based distributed manufacturing control: A state-of-the-art survey. Engineering Applications of Artificial Intelligence (2008)
14. Lind, M., Roulet-Dubonnet, O.: Holonic shop-floor application for handling, feeding, and transportation of workpieces. International Journal of Production Research 49, 1441–1454 (2011)
15. Lind, M., Skavhaug, A.: Using the blender game engine for real-time emulation of production devices. International Journal of Production Research, 1–17 (2011)
16. Makarenko, A., Brooks, A., Kaupp, T.: Orca: Components for robotics. In: International Conference on Intelligent Robots and Systems (IROS), pp. 163–168. IEEE (2006)

17. Monostori, L., Váncza, J., Kumara, S.: Agent-based systems for manufacturing. CIRP Annals-Manufacturing Technology 55(2), 697–720 (2006)
18. Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., Ng, A.: ROS: an open-source Robot Operating System. In: Open-Source Software Workshop of the International Conference on Robotics and Automation (ICRA). IEEE (2009)
19. Roulet-Dubonnet, O., Brøtan, V.: Controller integration in an industrial hybrid manufacturing cell. In: IWAMA 2012: International Workshop of Advanced Manufacturing and Automation. Tapir Academic Press (2012)
20. Roulet-Dubonnet, O., Lind, M.: IceHMS: A Holonic Middleware (2010), `https://github.com/oroulet/icehms/`
21. Roulet-Dubonnet, O., Nyen, P.A.: A method and application to simulate and validate manufacturing control systems based on a discrete manufacturing simulation platform. In: Mařík, V., Martinez Lastra, J.L., Skobelev, P. (eds.) HoloMAS 2013. LNCS (LNAI), vol. 8062, pp. 152–162. Springer, Heidelberg (2013)
22. Roulet-Dubonnet, O., Ystgaard, P.: An Application of the Holonic Manufacturing System to a Flexible Assembly Cell. In: Mařík, V., Vrba, P., Leitão, P. (eds.) HoloMAS 2011. LNCS, vol. 6867, pp. 29–38. Springer, Heidelberg (2011)
23. Scholz-Reiter, B., Freitag, M.: Autonomous processes in assembly systems. CIRP Annals - Manufacturing Technology 56(2), 712–729 (2007), `http://www.sciencedirect.com/science/article/B8CXH-4R68JFX-6/2/ec99c919c6a575b5517be08b071155ee`
24. Shen, W., Hao, Q., Yoon, H., Norrie, D.: Applications of agent-based systems in intelligent manufacturing: An updated review. Advanced Engineering Informatics 20(4), 415–431 (2006)
25. Soucek, S., Sauter, T.: Quality of service concerns in IP-based control systems. IEEE Transactions on Industrial Electronics 51(6), 1249–1258 (2004)
26. The Free Software Foundation: GNU General Public Licence (2011), `http://www.gnu.org/copyleft/gpl.html`
27. Vallejo, D., Albusac, J., Mateos, J., Glez-Morcillo, C., Jimenez, L.: A modern approach to multiagent development. The Journal of Systems & Software (2009)
28. Van Brussel, H.: Holonic Manufacturing Systems The Vision Matching the Problem. In: Proceedings of the 1st European Conference on Holonic Manufacturing Systems, Hannover, Germany, IFW-Hannover (1994)
29. Varela, C., Abalde, C., Castro, L., Gulías, J.: On modelling agent systems with erlang. In: Proceedings of the 2004 ACM SIGPLAN Workshop on Erlang, ERLANG 2004, pp. 65–70. ACM, New York (2004), `http://doi.acm.org/10.1145/1022471.1022481`
30. Weyns, D., Holvoet, T.: A formal model for situated multi-agent systems. Fundamenta Informaticae 63(2), 125–158 (2004)
31. ZeroC Inc.: Distributed Programming with Ice (2010), `http://zeroc.com/doc/latest/manual/`

Formal Study of Reconfigurable Manufacturing Systems: A High Level Petri Nets Based Approach

Laid Kahloul[1], Karim Djouani[2], and Allaoua Chaoui[3]

[1] LINFI Laboratory, Computer Science Department, Biskra University, Biskra, Algeria
[2] LISSI Laboratory, UPEC University, Paris, France
[3] MISC Laboratory, Department of Computer Science and its Applications, Faculty of NTIC, University Constantine 2, Algeria
l.kahloul@univ-biskra.dz, kahloul2006@yahoo.fr, djouani@gmail.com, a_chaoui2001@yahoo.com

Abstract. A Reconfigurable Manufacturing System (RMS) has the ability to adapt itself for new requirements and to handle accidental damages. This reconfiguration makes the system more efficient, thus, enhances its productivity. However, the development process of these systems and their reliability remain a big challenge. Formal methods are considered as one issue for designing theses systems and analysing their properties. The objective of this paper is to apply Petri Nets based methods in the modelling and simulation of RMS. In particular, RONs (Reconfigurable Object Nets) are used as a high level Petri Nets to catch the reconfigurability aspect in RMSs. This exploitation allows the developer to simulate, and verify formally the correctness and reliability of the RMSs.

1 Introduction

Manufacturing Systems (MSs) [1] are extensively used in industry. They are characterised by their hybrid aspect and their complexity. A manufacturing system is, usually, composed of a set of components: machines, robots, conveyers, buffers, and eventually humans. These components co-exist and interact to produce some products. The interaction is done through the exchanging information messages or through the circulating products during the manufacturing process. The success of the system is based on the quality of each component and also on the quality of these interactions. Reconfigurable manufacturing systems (RMSs) [2, 3] are MSs, where the components and their interaction can change over time. The structure of the system is no more static but dynamic. This reconfigurability makes the system more flexible and so allows an adaptation which enhances the productivity. In a reconfigurable manufacturing system, the flow changes dynamically and the components auto-reconfigure to answer new requirements or to handle damages.

Despite the advantages of reconfigurability, it makes the RMS more complex and its development becomes a hard task. Reconfigurability brings new challenges and new types of errors can appear. These errors require a sophisticated verification process to insure the reliability of RMSs. The Verification of RMSs can be done using classical techniques used for manufacturing systems. The Petri Net (PN) formalism [4] is one of

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 106–117, 2013.

the most used tools in the study of discrete event systems. This formalism attracted, early, an important interest in the study of manufacturing systems [5-8]. However, with its classical definition, as a non dynamic formalism, it seems to be not suitable to catch the new reconfiguration aspect in RMSs. New extensions of Petri Nets were proposed to deal with reconfiguration and, in general, for the study of dynamic systems [9-13].

In this paper, we propose to use Reconfigurable Object Nets (RONs) [14] to deal with the formal modelling of RMSs. RONs are high level Petri Nets, where the tokens can be nets (called token nets). The token nets can change their structure due to reconfiguration rules (modelled also as other tokens in the RON). The mathematical background of reconfiguration in RONs is based on graph transformation theories [15]. The second advantage of the RONs formalism resides in its implementation in some automatic tools like the RON-tool [16]. These tools allow the automatic modelling and simulation of systems.

This paper is organised as follows: Section 2 details the Reconfigurable Object Nets formalism and its background in graph transformation techniques. Section 3 presents a case study of an RMS (Reconfigurable Manufacturing System) and discusses the approach followed to create the RON model for the presented case study. Section 4 presents some related works. Finally, section 5 concludes the paper and draws some perspectives of the work.

2 Reconfigurable Object Nets

2.1 An Informal Presentation

RONs [24] were introduced firstly in [14] as High Level Nets with Nets and Rules as Tokens. In RONs, one can distinguish between two levels in the Net: the system level and the token Level. Tokens have two kinds: token nets and token rules. Thus, places in the system level can contain nets or rules. A token net is a P/T net which can move from one place to another place in the system. When moving, a token net's marking can change as well as its structure. The transitions in the system level decide about the movement of token nets from one place to another. These transitions decide if the marking or the structure of a token net must be changed. To change the marking of a token net, the transition in the system level triggers a transition in the token net level. However, to change the structure of a token net, a transition, in the system level, requires a token rule. The token rule decides how the structure of the token net must be changed when some transition, in the system level, is fired. In RONs, reconfiguration of the structure concerns only the token nets and not the system level. This reconfiguration is defined by a set of token rules, based on the graph transformation techniques.

Transformation techniques inspired from graph transformation allow the formulation of two basic constructions: **union** (or **gluing**) and **transformation** on Place/Transition Nets (P/T nets). The **union** construction takes two Nets N_1 and N_2 and yields another net N_3 while the **transformation** construction takes one P/T net N_1 and yields another net N_2. In this approach, these two constructions are the two basic ***reconfigurable*** techniques for P/T nets. Union and transformation are based on the **morphism** concept defined over P/T nets. Bellow, we present Place/Transition nets, morphisms over P/T nets, union, transformation, and finally RONs.

2.2 Place/Transition Nets (P/T nets)

A place/transition net is a quadruplet (T, P, *Pre, post*), where:

- T : a finite set of transitions ;
- P : a finite set of places;
- *Pre* (for pre-domain) and *Post* (for post-domain) are two mappings defined as *Pre, Post*: $T \rightarrow P^{\oplus}$.

The set $P^{\oplus}$ is the set of finite multi-sets over the set P. An element *w* in $P^{\oplus}$ can be written as the sum: $w=\sum_{p\in P} \lambda_p \times p$. The λ_p is a natural number ($\lambda_p \in \mathbb{N}$). *w* can also be written as a function: $w: P \rightarrow \mathbb{N}$.

2.3 Morphisms over P/T nets

Given two P/T nets: $N_1 = (T_1, P_1, Pre_1, Post_1)$, and $N_2 = (T_2, P_2, Pre_2, Post_2)$, a **morphism** *f between* the two nets N_1 and N_2 is a function $f: N_1 \rightarrow N_2$. We have $f = (f_T, f_P)$, such that: $f_T : T_1 \rightarrow T_2$, and $f_P: P_1 \rightarrow P_2$ are two morphisms which map transitions into transition and places into places, respectively. f_T and f_P satisfy:

1) $Pre_2 \circ f_T = f_P^{\oplus} \circ Pre_1$
2) $Post_2 \circ f_T = f_P^{\oplus} \circ Post_1$

The diagram (Fig. 1 (a)) [14] summarizes the above concepts.

2.4 Union of P/T Nets as a Pushout

Based on the morphisms on P/T nets, it was possible to define a specific construction which is the **pushout (or union)** of two P/T nets. Let $N_1 = (T_1, P_1, Pre_1, Post_1)$, $N_2 = (T_2, P_2, Pre_2, Post_2)$, and $I = (T_0, P_0, Pre_0, Post_0)$ be three nets, with the two morphisms: $f: I \rightarrow N_1$ and $g: I \rightarrow N_2$. The net I is said a common interface between N_1 and N_2. The union of N_1 and N_2 is the Net N= (T, P, *Pre, Post*), defined using the two morphisms f': $N_1 \rightarrow N$ and g': $N_2 \rightarrow N$. We write $N = N_1 +_I N_2$. The operator $+_I$ is called the **pushout** construction (Fig. 1 (b)) or the **gluing** operator.

The net N is constructed as follows:

1) $T=T_1+_{T0}T_2$. T is the disjoint union of T_1 and T_2, where we glue together transitions t, such that: $f_T(t) \in N_1$ and $g_T(t) \in N_2$ for each $t \in T_0$.

2) $P=P_1+_{P0}P_2$. P is the disjoint union of P_1 and P_2, where we glue together places p such that: $f_P(p) \in P_1$ and $g_P(p) \in P_2$ for each $p \in P_0$;

3) $Pre(t) = Pre_1(t_1)$ if $g'_T(t_1)=t$ else $Pre(t) = Pre_2(t_2)$ if $f'_T(t_2)=t$. The set of input places of a transition t in the new net N is the set of the input places of the transition t_1 (resp. t_2) defined in N_1 (rsp. in N_2) iff t is the image of t_1 (resp. of t_2) by the morphism g' (resp. by the morphism f').

4) $Post(t) = Post_1(t_1)$ if $g'_T(\underline{t_1})=t$ else $Post_2(t_2)$ if $f'_T(t_2)=t$. The set of output places of a transition t in the new net N is the set of the input places of the transition t_1 (resp. t_2) defined in N_1 (rsp. in N_2) iff t is the image of t_1 (resp. of t_2) by the morphism g' (resp. by the morphism f').

2.5 Transformations of P/T Nets as a Double Pushout Rules

Based on the P/T gluing construction, the P/T transformation is constructed as a double pushout. Let L, K, R, and C be four P/T nets. A transformation "*f*: $N_1 \rightarrow N_2$" transforms the P/T net N_1 to the P/T net N_2 using the rule *r*=(L, K, R) and the matching *m*: $L \rightarrow N_1$ iff we have the double pushout of the Fig. 1. (c).

In the Fig. 1(c), k_1, k_2, c, n are morphisms. The P/T net C is called the **context** of transformation, and it satisfies:

1) $T_C = (T_1 \backslash m_T(T_L)) \cup m_T(k_{1T}(T_K))$. Transitions of C are the **union** of the set of transitions in N_1 (excluding those which are images of L's transitions by m) and the set of K's transitions images by the morphisms composition: $(m \circ k_1)$;
2) $P_C = (P_1 \backslash m_P(P_L)) \cup m_P(k_{1P}(P_K))$. Places of C are the **union** of the set of Places in N_1 (excluding those which are images of L's places by m) and the set of K's places images by the morphisms composition: $(m \circ k_1)$;
3) $Pre_C = Pre_{1|TC}$. (The relation Pre_C is the same relation Pre_1 restricted only to the set T_C)
4) $Post_C = Post_{1|TC}$. (The relation $Post_C$ is the same relation $Post_1$ restricted only to the set T_C)

$$\begin{tikzcd} T_1 \arrow[r, shift left, "pre_1"] \arrow[r, shift right, "post_1"'] \arrow[d, "f_T"] & P_1^{\oplus} \arrow[d, "f_P^{\oplus}"] \\ T_2 \arrow[r, shift left, "pre_2"] \arrow[r, shift right, "post_2"'] & P_2^{\oplus} \end{tikzcd}$$

(a)

$$\begin{tikzcd} I \arrow[r, "f"] \arrow[d, "g"'] \arrow[dr, phantom, "(1)"] & N_1 \arrow[d, "g'"] \\ N_2 \arrow[r, "f'"'] & N \end{tikzcd}$$

(b)

$$\begin{tikzcd} L \arrow[d, "m"'] \arrow[dr, phantom, "(1)"] & K \arrow[l, "k_1"'] \arrow[r, "k_2"] \arrow[d, "c"] \arrow[dr, phantom, "(2)"] & R \arrow[d, "n"] \\ N_1 & C \arrow[l] \arrow[r] & N_2 \end{tikzcd}$$

(c)

Fig. 1. (a) Morphisms on P/T nets, (b) Union of P/T nets, (c) Double pushout

3 Modelling and Simulation of Reconfigurable Manufacturing Systems

3.1 The Case Study

We consider brickwork for manufacturing of clay products. This system (Fig. 2) is composed of three manufacturing cells (MC1, MC2, MC3), and an AVG (automated guided vehicle). Initially the system produces brick building only. The system inputs are two raw materials water and red clay. The flow starts in MC1, then passes to MC2, and finally finishes in MC3.

MC1 contains a filtering machine (FM) to filter the clay, and a mixing machine (MM) to mix water with red clay. The output of the MC1 is a clay dough to be passed to MC2. MC2 contains an extrusion machine (EM) which makes the form of the bricks, and a cutting machine (CM). Finally, MC3 contains a dryer (DR), an oven (OV), and a drilling machine (DM). In each manufacturing cell, a set of conveyers (CNV) or robots are used to transport intermediary products, and a set of buffers are used to temporary save these intermediary products .

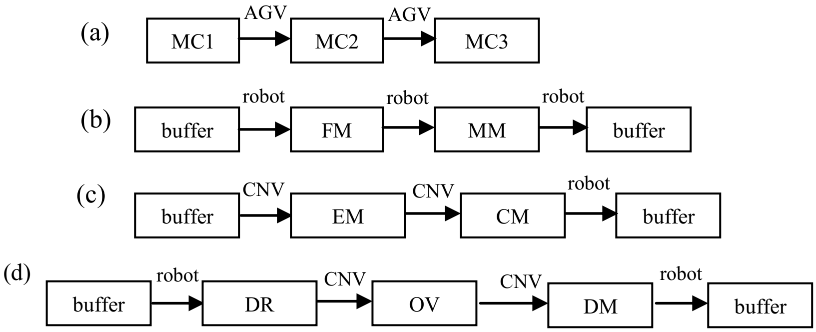


Fig. 2. (a) Manufacturing chain, (b) Flow in MC1, (c) Flow in MC2, (d) Flow in MC3

We propose that the system will be updated (reconfigured) to satisfy a new requirement, which is the production of a new kind of bricks (used in the slabs). A new extrusion machine (EMS) is introduced and a new cutting machine (CMS) is introduced in the system. The system will produce two kinds of products according to controller command. The new design of MC2 will be as shown in Fig. 3.

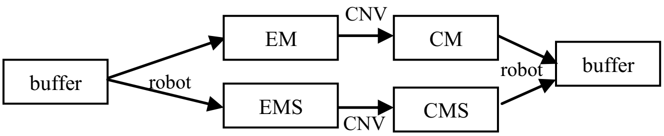


Fig. 3. The new flow in MC2

3.2 The Modelling Process

The modelling using RON requires the definition of the two levels: **System Level** and **Token Level**. In the Token level, one must identify: the set of token nets (P/T nets which describe the structure and the behaviour of the manufacturing system), and the set of token rules (production double pushout rules, which describe the reconfigurations that can be applied on the manufacturing system's structure). In the system level, places can be net places (contain token nets) or rule places (contain token rules). The transitions, in the system level, have the ability to trigger the transitions in a token net, so that they change the token net marking. In this case, transitions (of level system) are called **Fire** Transitions. A second ability is to change the token net structure by the application of token rule. In this case, transitions (of level system) are called T**ransform** Transitions.

Identification of Token Nets

Two tokens nets (TNs) are defined (Fig. 4). These two nets represent the two configurations of the system, during its execution. The interpretation of the set of nodes, in these token nets, is presented on table I.

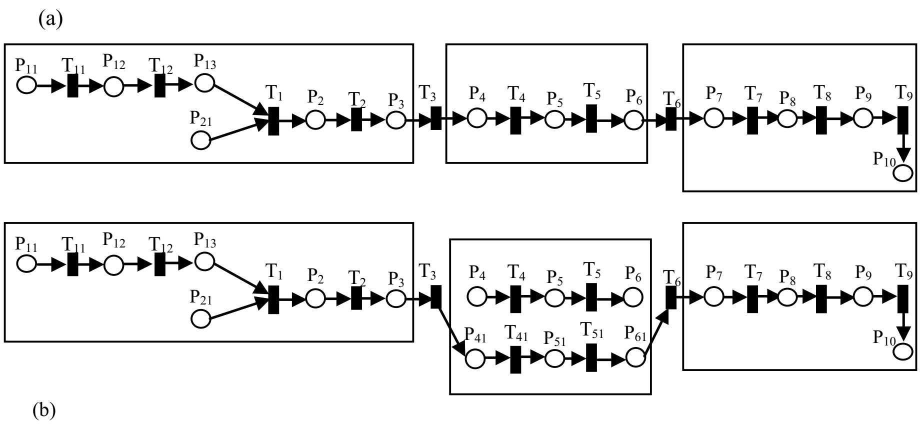


Fig. 4. (a) TN_1: for the initial brick manufacturing process, (b) TN_2: for the system after reconfiguration

Table 1. Interpretation of Nodes in Token Nets

node	***Interpretation in the manufacturing system***	***node***	***Interpretation in the manufacturing system***
P_{11}, P_{12}, P_3	Buffers (for red clay, water, final product) in MC1	T_7	DR (in MC3)
(T_{11}, P_{12}), P_{13}, (P_2, T_2)	robots (in MC1)	P_8, P_9	The two CNVs (in MC3)
T_{12}	FM(in MC1)	T_8	OV (in MC3)
T_1	MM (in MC1)	T_9	OV (in MC3)
P_4	Buffer and first CNV (in MC2)	P_{10}	Second Robot and last Buffer (in MC3)
T_4	EM (in MC2)	P_{41}	Buffer and first CNV (in MC2 : second configuration)
P_5	Second CNV (in MC2)	T_{41}	EMS (in MC2: second configuration)
T_5	CM (in MC2)	P_{51}	Second CNV (in MC2: second configuration)
P_6	Robt and final buffer (in MC2)	T_{51}	CMS (in MC2: second configuration)
T_3, T_6	AVG	P_{61}	Robot and final buffer (in MC2: second configuration)
P_7	Buffer and first robot (in MC3)		

Identification of Token Rules

A production rule must be defined, which will trigger the reconfiguration process of the manufacturing system. The construction of this rule requires the definition of a set of morphisms. On Fig. 5, we depict the production rule p= (L, I, R), where (L: for left, I: for Interface, and R: for Right).

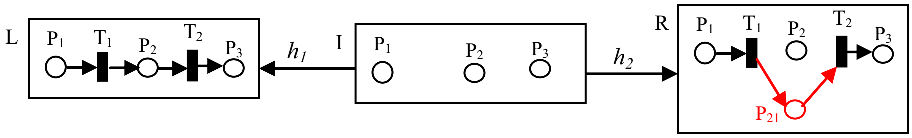


Fig. 5. The production rule

The two morphisms h_1 and h_2 are depicted on the figure Fig. 6.

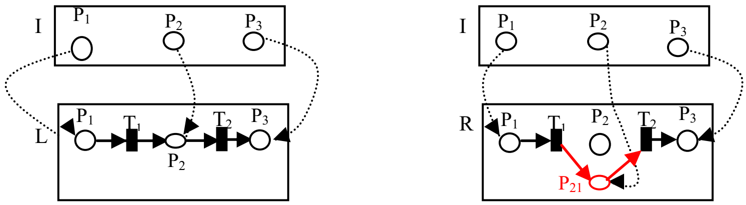


Fig. 6. Morphisms h_1 and h_2

Now, the double pushout rule is depicted on the Fig 7. On Fig. 7, h_{11}, h_{12}, c, m, g are morphisms, and C is a context net. Once the morphism m is defined the context C can be computed using the definition presented above in the subsection (2.5). The double push out rule is r_1=(p,m), and we have the transformation: $TN_1 \rightarrow^{(p,m)} TN_2$.

$$\begin{tikzcd} L \arrow[d, "m"'] & I \arrow[l, "h_1"'] \arrow[r, "h_2"] \arrow[d, "c"] & R \arrow[d, "g"] \\ TN_1 & C \arrow[l, "h_{11}"'] \arrow[r, "h_{12}"] & TN_2 \end{tikzcd}$$

Fig. 7. Double push out for rule

The figures Fig. 8 and Fig. 9 present in details the two morphisms m and g. It is easy to test that the two relations m and g satisfy requirements to be morphisms.

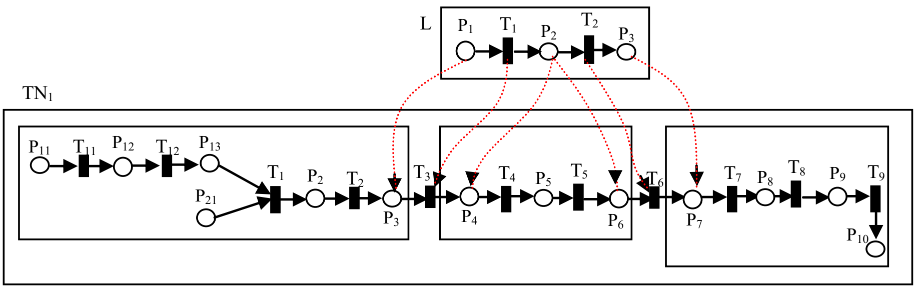


Fig. 8. The morphism m

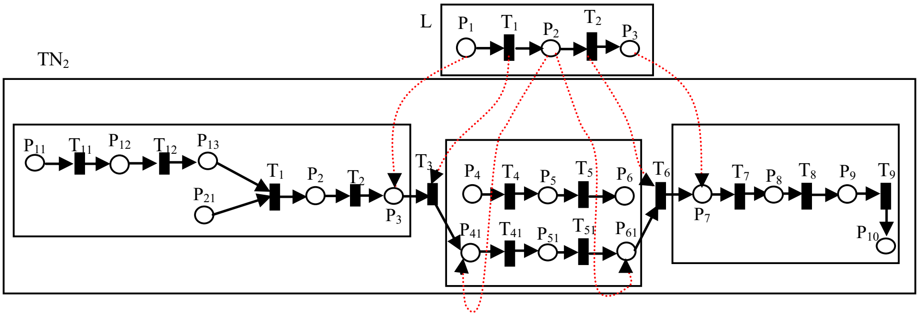


Fig. 9. The morphism g

Now, we can compute the context C, using the definition presented above in the section 2.5.

The System Level Net. In the system level, we must have the set of transitions which trigger the dynamic behaviour (**Fire Transitions**) and reconfigurations (**Transform Transitions**) in the token nets. The tokens nets are the markings of the places in the system level.

A Fire Transition takes a net *N*, a transition *t* in this net, and update the marking of N_1 by firing *t* (if this last one is enabled). A Fire Transition must have a guard [*enabled(t)=true*]. It produces a new net: *fire(N, t)*.

A Transform Transition takes a net *N*, a rule *r*=(p, m), and apply this rule to transform *N*. A transform transition must have a guard [*applicable(N,r)*]. It produces a new net with a new structure defined by a function *transform(N, r)*.

On Fig. 10, we depict the Reconfigurable Object Net for the system described in this paper. Each place (a circle) has a name (depicted near the circle, in the right high side), a type (depicted near the circle, in the right low side), and an eventually initial marking (inside the circle). Each transition (a rectangle) has a name (depicted inside the rectangle), and eventually guard (depicted outside the rectangle). An arc links a place to a transition or a transition to a place, and has a label depicted near it.

In the RON of Fig. 10, the place np_1 is initially marked by the token net TN_1 (the initial configuration of the reconfigurable system). The place rp_1 is initially marked r_1.

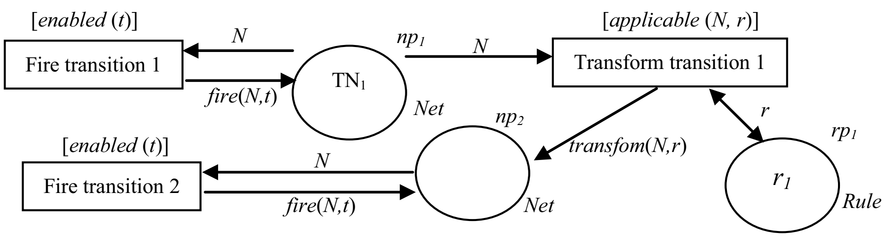


Fig. 10. The RON model of the system

3.3 Simulation Using the RON-Tool

One of the advantages when using the RON formalism is the ability to simulate the model with the RON-tool [16]. The RON-tool is free and can be downloaded (with its open source). The current version allows, only, the simulation of the model. The verification of properties is not yet implemented [16]. However, the availability of the source allows the implementation and the specialisation of the verification process by designers.

4 Related Works

In this section, we examine some recent works which are closed to our work. We can classify these works into two principal categories: works where PN (without dynamic structure) are applied to RMS [17-22], and works which uses High level Petri nets (with dynamic structure) as in [22, 23].

The first category of works finds its motivation in the maturity of used formalisms: p-time PN in [17], Coloured PN in [18], and Coloured PN in [18, 19, 20]. In this category of extensions, some works treated the reconfiguration in a modular way to facilitate the building of new models after reconfiguration. They enrich PN with oriented object concepts (derivation, inheritance) or modularity concept to overcome the reconfigurability complexity. These ideas can be found in [21, 22]. Authors of [21] used coloured timed oriented object to facilitate reconfiguration of the PN models. Where the Petri nets models are seen as objects in classes, and new objects can be derived from other objects. This derivation can be seen as a reconfiguration. In [22], the authors proposed ITPN (Intelligent Token Petri Nets). In ITPN, tokens are enriched with time, and knowledge about which transitions must be disabled when these tokens are consumed. A synthesis process is proposed to construct new nets from other nets. This process facilitates the definition of new models from existing ones. However, no mechanism is included in the ITPN to realise this reconfiguration. Thus, the dynamic of the structure is not implemented in the net itself. Finally, we can say that the power of these models resides in the existence of well defined analysis techniques, where many properties are decidable. Many automatic tools are proposed to model, simulate and analyse systems using these formalisms. The major lack in these approaches is the absence of ability to represent explicitly and intuitively reconfiguration of the system. In our work, we are interested to use formalisms where the reconfiguration of the system can be modelled, explicitly, through the dynamic structure of the formalism. Thus, our work can be inscribed in the second category of works.

In the second category of works, the PN model is enriched by some mechanism to reconfigure itself. Thus, the model is more intuitive and natural to support the modelling of Reconfigurable Manufacturing Systems (RMS). Probably, one of the first works where the reconfiguration in PNs was used to model Manufacturing Systems can be found in [15]. However, the context of the study was the graph transformation, and the reconfigurability in PNs was used to model a refinement development process of manufacturing systems, and not a reconfigurable manufacturing system. The same idea was used after to develop Reconfigurable Petri Nets [24], and RONs (Reconfigurable Object Nets) [14]. Li et al. developed Improved net rewriting systems

(INRS) in [22], which are initially based on Badouel's reconfigurable Petri nets [10]. They propose a hybrid approach based on UML.2 and INRS, to design RMS. In [23], authors propose hybrid reconfigurable Petri nets as a new formalism which combines hybrid aspect (continue and discreet modelling) and reconfiguration aspect. However the used model is not yet developed and no results about the verification are available. At our knowledge, there is no automatic tool for the verification of INRS model properties neither for HRPN. The use of RONs finds also its motivation in the availability of the automatic tool: RON-tool [16] which allows editing graphically and simulating of the model. Even the available version is not equipped with verification techniques; the availability of the "source code" helps designers in the implementation of verifications techniques.

5 Conclusion

Reconfigurable manufacturing systems (RMSs) find their application in many domains. These systems offer high flexibility to the production systems. They can adapt themselves to satisfy new requirements or to avoid accidental damages. The design of these systems represents a challenge for researchers. The use of formal methods brings important solutions. The formal specification allows designer to prove some inherent properties for these systems: deadlock free, reachability analysis, reliability, liveness, …

In this paper, we have presented an experience where the Reconfigurable Object Nets are used to specify a RMS (Reconfigurable Manufacturing System). This formalism is based on graph transformation techniques. In this paper, we have examined a case study, where the RMS encounters one reconfiguration (to satisfy a new required product). This reconfiguration is specified as a transformation rule to be applied on the structure of the system. We have presented the final model as a RON (Reconfigurable Object Net). This work opens many perspectives. We propose to develop this work on two levels: (i) enrich the work and develop a concrete approach that can be used in the modelling of RMSs using RONs, (ii) working on the RONs automatic tool (open source), to implement properties verification processes. We plan also to use graph transformation tools such as TGG to implement the transformation of RONs models to equivalent Petri nets models for which verification of properties exist.

References

1. Serope, K., Schmid, S.: Manufacturing, Engineering & Technology, pp. 22–36, 951–988. Prentice Hall (August 2005) ISBN 0-13-148965-8
2. Mehrabi, M.G., Ulsoy, A.G., Koren, Y.: Reconfigurable manufacturing systems: key to future manufacturing. Journal of Intelligent Manufacturing 11(4), 403–419 (2000)
3. Katz, R.: Design principles of reconfigurable machines. The International Journal of Advanced Manufacturing Technology 34(5), 430–439 (2007)
4. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE 77(4), 541–580 (1989)

5. Valette, R., Cardoso, J., Dubois, D.: Monitoring manufacturing systems by means of Petri nets with imprecise markings. In: IEEE International Symposium on Intelligent Control, September 25-26, pp. 233–238 (1989)
6. Zhou, M.C., Mcdermott, K., Patel, P.A.: Petri net synthesis and analysis of a flexible manufacturing system cell. IEEE Transactions on Systems, Man, and Cybernetics 23(2), 523–531 (1993)
7. Cheng, C.W., Sun, T.H., Fu, L.C.: Petri-net based modeling and scheduling of a flexible manufacturing system. In: IEEE International Conference on Robotics and Automation, pp. 513–518 (1994)
8. Wang, L.C.: Object-oriented Petri nets for modeling and analysis of automated manufacturing systems. Computer Integrated Manufacturing Systems 26(2), 111–125 (1996)
9. Valk, R.: Self-modifying nets, a natural extension of petri nets. In: Ausiello, G., Böhm, C. (eds.) ICALP 1978. LNCS, vol. 62, pp. 464–476. Springer, Heidelberg (1978)
10. Badouel, E., Oliver, J.: Reconfigurable nets: a class of high level petri nets supporting dynamic changes with workflow systems. Research Report Publication 1163, Irisa (Institut de Recherche en Informatique et Systmes Aleatoires) (1998)
11. Valk, R.: Petri Nets as Token Objects: An Introduction to Elementary Object Nets. In: Desel, J., Silva, M. (eds.) ICATPN 1998. LNCS, vol. 1420, pp. 1–25. Springer, Heidelberg (1998)
12. Lomazova, I.A.: Nested Petri Nets: Multi-level and Recursive Systems. Fundamenta Informaticae 47(3), 283–293 (2001)
13. Asperti, A., Busi, N.: Mobile Petri Nets. Mathematical Structures in Computer Science Journal 19(6), 1265–1278 (2009)
14. Hoffmann, K., Ehrig, H., Mossakowski, T.: High-Level Nets with Nets and Rules as Tokens. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 268–288. Springer, Heidelberg (2005)
15. Ehrig, H., Padberg, J.: Graph Grammars and Petri Net Transformations. In: Desel, J., Reisig, W., Rozenberg, G. (eds.) ACPN 2003. LNCS, vol. 3098, pp. 496–536. Springer, Heidelberg (2004)
16. Biermann, E., Ermel, C., Hermann, F., Modica, T.: A Visual Editor for Reconfigurable Object Nets based on the E CLIPSE Graphical Editor Framework. In: Proc. 14th Workshop on Algorithms and Tools for Petri Nets (AWPN 2007). GI Special Interest Group on Petri Nets and Related System Models, vol. 10 (2007)
17. Julia, S., de Oliveira, F.F., Valette, R.: Real time scheduling of Workflow Management Systems based on a p-time Petri net model with hybrid resources. Simulation Modelling Practice and Theory 16(4), 462–482 (2008)
18. Zhang, L.L., Rodrigues, B.: A Petri Net-based Approach to Reconfigurable Manufacturing Systems Modeling. Research Collection Lee Kong Chian School of Business (Open Access), Paper 3047
19. Meng, X.: Modeling of reconfigurable manufacturing systems based on colored timed object-oriented Petri nets. Journal of Manufacturing Systems 29(2-3), 81–90 (2010)
20. de Aguiar, C., José, A., Villani, E., Junqueira, F.: Coloured Petri nets and graphical simulation for the validation of a robotic cell in aircraft industry. Robotics and Computer-Integrated Manufacturing 27(5), 929–941 (2011)
21. Li, J., Dai, X., Meng, Z., Dou, J., Guan, X.: Rapid design and reconfiguration of Petri net models for reconfigurable manufacturing cells with improved net rewriting systems and activity diagrams. Computers & Industrial Engineering 57(4), 1431–1451 (2009)

22. Wu, N., Zhou, M.: Intelligent token Petri nets for modelling and control of reconfigurable automated manufacturing systems with dynamical changes. Transactions of the Institute of Measurement and Control 33(1), 9–29 (2009)
23. Lejri, O., Tagina, M.: Hybrid Reconfigurable Petri Nets for modelling Hybrid Reconfigurable Manufacturing Systems. Journal of Studies on Manufacturing 1(2-3), 75–84 (2011)
24. Biermann, E., Modica, T.: Independence Analysis of Firing and Rule-based Net Transformations in Reconfigurable Object Nets. In: Electronic Communications of the EASST Seventh International Workshop on Graph Transformation and Visual Modeling Techniques, vol. 10 (2008), http://www.easst.org/eceasst/

A Virtual Design Engineering Environment for Control System Design and Validation

Francisco P. Maturana[1], Maulik Kalolia[2], and Polo Paredes[1]

[1] Advanced Technology Lab, Rockwell Automation, Ohio, USA
{fpmaturana,lparedes}@ra.rockwell.com
[2] Fenn College of Engineering, Cleveland State University, Ohio, USA
kaloliamaulik@gmail.com

Abstract. Companies are looking for a more efficient way to design, validate, test and commission a control system. In near future, automated intelligence (Agents/Holons) that will support the control logic actions will need some form of environment for easing the validation of the decision making rules. This paper discusses tools to help increase design efficiency and productivity, by enabling virtual multi-disciplinary engineering collaboration through an integrated set of design software products. This article proposes an integration of information from different design, simulation, control and visualization systems into a common platform for simulation and control validation. The proposed technology provides an analytical framework for system design, including virtualized control technology connected to a simulation engine coupled with 3D visualization. The capability of this prototype to visualize real time control of mechanisms in a virtual environment, proves how a virtual design engineering environment for control system design and validation, can help qualify the production controller before connecting it to the real machine. It will also assist in understanding the behavior of the system, detect problems early in the design process, reduce downtime, troubleshooting and project cost. These tools will help reduce development time of control systems and improve productivity.

Keywords: agents, control, design, model, simulation, validation, virtual.

1 Introduction

Presently, advanced product development methods are used in industrial manufacturing technology to reduce product design cycle time and sustain in the market competition. More complex products are designed easily using sophisticated CAD design tools. These CAD designs contain high level model information like body mass, inertia, stiffness, and tension. These models also contain information about the integrity between different parts, clearance, joint movement, etc. All these parameters can be defined in the CAD assembly drawings. Physical prototypes are made from these conceptual 3D designs. Industrial Programmable Logic Controllers (PLCs) are used to control these designed prototype mechanisms once built. Control algorithms are designed for the required automation applications. Traditional system engineering uses

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 118–127, 2013.

physical prototypes to validate various aspects of automation solutions. Sometimes this process requires costly hardware to validate control algorithms without having physical prototypes. If some problems occur during the physical prototyping, there are fewer chances to modify the design. This aspect of design introduces a limiting degrees-of-freedom into the product development cycle and potentially affects the time-to-market of the final product. Moreover, evidence indicates that classical control systems cannot longer stand alone in controlling process and machines. They need automated intelligence supporting their operations. The combined action of pure control with agents will make the industrial system more complicated to validate at design. Thus the need for thinking about a virtual design engineering environment comes to the table.

A survey on product development methods, design cycle, product cost was done by Aberdeen group, 2008 among 140 companies [1]. It shows that 32% of the surveyed companies faced difficulties in predicting and modeling system behavior until physical prototype existed. The survey also suggested a requirement for a work flow to integrate mechanical, electrical, and software components to shorten product development schedules. A model-based simulation and visualization system is then required to validate control algorithms early in the process while designing the components. In the control engineering domain, controlling mechanical systems like robots, conveyors, or sliding mechanisms require physical models to validate system parameters like position, torque, and speed based on variable demand and throughput. To reduce the product development cycle time, a flexible manufacturing system is required, in which design and control programs can be easily tuned. Such a system demands for model based simulation and visualization that will help to evaluate dynamical aspects of the components under load and to visualize the mechanical response of the machine on the fly, earlier in the design lifecycle. This is the theory of a rapid prototyping control validation system. Control program algorithms can be applied to the simulation model and the system response can be feedback to controls and viewed using 3D virtual models. Previous art in this field has been reported in [9].

Different tools and software packages are used in each phase of product development for design, control and simulation. Each tool has different data types and information exchange requirements. The objective of this paper is to develop a mechatronics-based system to integrate design, simulation, controllers, and visualization into a development environment. This environment is intended to smoothening transition from the design cycle into the runtime validation and to rapidly evaluate the impact of the design decisions into the real time model of the machine. The intention of this work is to materialize the theory into a practical environment that would help designers and engineers with a platform for product development. To create this kind of environment, the following requirements have been envisioned:

- Establish a data exchange interface in between controller and simulation to enable system control and system response evaluation,
- Define model-based simulations to predict the system behavior, and
- Convert CAD model information into simulation and animation components.

Step-by-step development of application and interface of the above requirements are discussed in this paper. The remainder of the paper is structured as follows. Section 2 provides an overview of the system architecture and procedures. Section 3 will show the benefit of this application applied to a robot control application example. Future development and conclusions are mentioned in section 4.

2 System Architecture

To integrate design, control, simulation and visualization, a system work flow is proposed. Figure 1 shows the general composition of the system architecture, consisting of five main parts: 1) Control program design, 2) Controllers, 3) Plant Simulation, 4) CAD, and 5) Visualization. Industry based tools and open source software has been selected to carry out these design activities. For example, control design is carried in the Rockwell Automation's Studio5000programming tool. Control program execution is carried out in the Emulate virtual controller from Rockwell Automation. Component/Machine simulation is carried out in the Matlab/Simulink™ tool. Solid modeling is carried in the Solidworks™ tool. And, visualization is performed in an open source Virtual Reality Modeling Language (VRML) tool from InstantReality company [2]. Each of these tools brings its own data structure and information formats into the design process. The definition of the workflow is the goal but the integration of the information sources and their respective formats is a requirement. The challenge for the integrated design and validation environment is to glue the pieces of information into a common information-exchange structure that can be used to harmonize the components into a rapid prototyping environment.

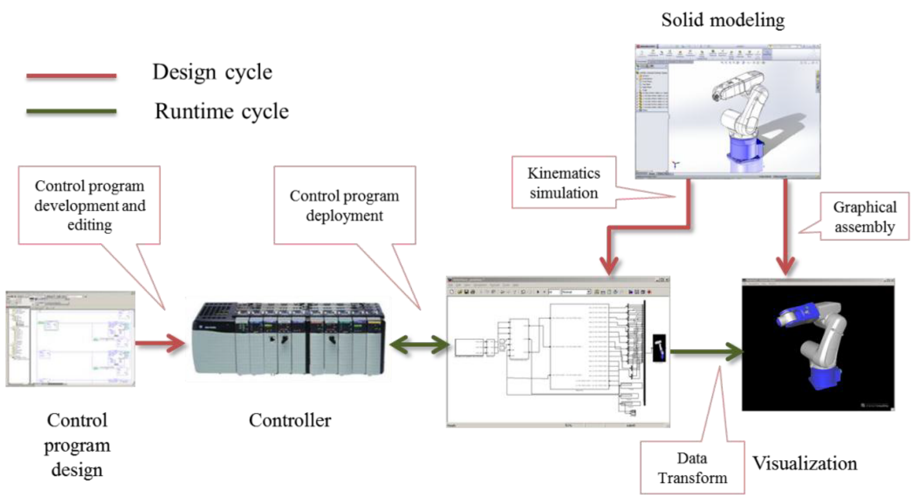


Fig. 1. System architecture

In Figure 1, a first information flow of the integrated system is shown. The solid modeling phase produces a three dimensional representation of the mechanical system to be controlled. The solid model is converted into two information elements: 1) kinematics simulation and 2) graphical assemblies. The Solidworks™ tool provides plugins to generate both data files as a Matlab™ simulation and VRML export files. The Studio5000™ tool is an industrial control design tool that is used to create the control algorithms written in an IEC 61131 based programming language. The control program is deployed in the virtual controller. The kinematics simulation is downloaded to the Matlab™ tool for simulating the machine under specific load conditions. The simulation is to interact with the controller at runtime to stimulate the controlling algorithms with the responses of the machine that is put under stress with dynamic loading and control profiles. Graphical assemblies will be further processed to become a hierarchical data structure of graphical description that can be animated in the visualization tool. The visualization tool collects the real time data that is generated in the controller-to-simulation interaction.

Different methods are proposed to connect each phase of this information flow. Control program parameters are transferred between the simulation and the controllers using an I/O interface. Translating CAD model information into Matlab/Simulink™ method was shown by Arkadiy and Terry [3]. Matlab™ provides SimMechanicsLinks plugin for CAD software Solidworks™ to transfer the CAD model into simulation. The simulation model contains mass and inertial properties of each body parts of the CAD drawings. This information is fundamental to be able to generate animation of the mechanical system. Kinematics and dynamical information of the model can be used to generate realistic model conditions since the actual physical properties of the model are included in the simulation calculations. A high quality and realistic animation can be produced in any VRML compatible tool using the exported VRML models. VRML provides open source, platform independent graphical language to create and view 3D models. VRML is a text based language so it can be easily edited using any text editor [4]. Limitations of CAD model's use and advantages of VRML based CAD assembly models were shown in [5] [6]. However, to bring the VRML tool into this workflow, it is necessary to attach an API to it so it can communicate with the other parts of the system.

Based on this work flow, an application can be created using these different systems under one platform. A test case is discussed here to show the use and benefits of this concept. For example, robots are widely used in industrial automation applications and their control algorithms are developed by system engineers. In our example, Denso VS-6556G six-axis articulated robot model is used [7] to proof-of-concept. A desired motion profile for the robot end effector position is established in the controlling program using command position profiles. The command positions are points of desired motion trajectory for the end effector. To achieve a desired position of the end effector, different parts (links) of the robot must be placed in a position by the driving motors at the link joints. These motors are driven by torque signals that are to be calculated by the controlling drivers. But the controlling torque depends on the knowledge of the system response to the torque and the load at the end effector. In the classical design approach, these pieces of information are known by trial and error or by experienced designers. The intent here is to capture the information flow for automating the process of generating the feedback information for the drivers and controllers. Based on the drivers' torque signal, each joint will move and the end effector will move in the space to a target location. Current position is the feedback to the drivers and controllers for close loop control. A step-by-step procedure to achieve such information exchange is discussed in next sections.

2.1 Translating Design Model in to Simulation

Figure 2 shows the solid model assembly of Denso VS-6556G six-axis articulated robot. This robot assembly contains six body parts and they are connected with 5 concentric type joint configurations. Each assembly joint will produce revolute motion between two parts and it represents rotary motor actuators. Assembly joint information helps the application engineer to configure actuator parameters for sizing the driver configuration. End effectors position and orientation will be controlled by all joint actuator movements based on kinematics calculations.

The simulation model that is created from the 3D CAD model in Solidworks™ contains body blocks and assembly parts that are connected using joint blocks. This simulation model is intended to represents the dynamic behavior of a designed 3D mechanical system. It can be used for control system development and verification. The control signals that are generated in the controller are connected to the joint blocks using port signals into joint actuator blocks from the SimMechanics toolbox of the Simulink™ tool, as shown in Figure 2.

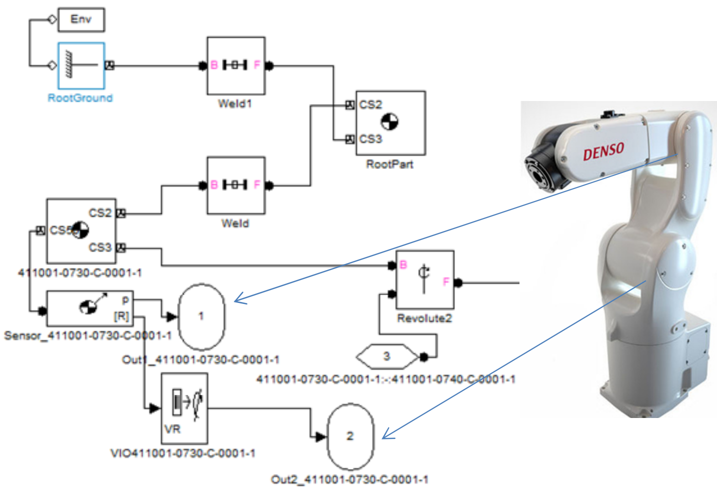


Fig. 2. CAD assembly converted to Solidworks™ format

2.2 Simulation and Controller Interface

To connect the virtual controller outputs signals to the simulation, it is necessary to establish an interface between simulation and controller. The SimKit tool that has been developed by the Advanced Technology Lab at Rockwell Automation enables connectivity and data exchange in a bidirectional way between simulation and controller. SimKit uses the Emulate controller to encapsulate the control programs. The emulator mimics the ControlLogix PLC functions but in software only, as shown in Figure 3.

The virtual controller transfers data to the simulation using output tags and receives data from the simulation using input tags. A Configurator component hosts the application-level information for connecting the virtual controller with the simulation. Tag Server coordinates the input and output tags data exchange. Synchronizer coordinates the clock progression between the controller and simulation to keep them synchronized.

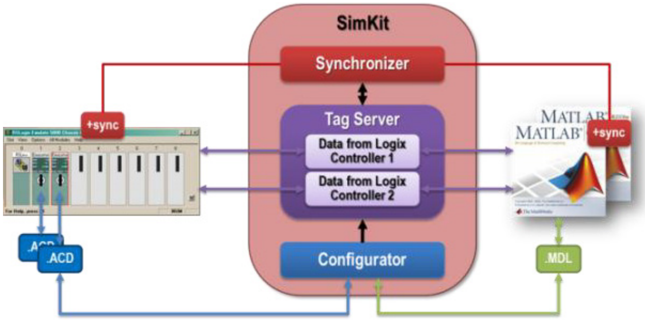


Fig. 3. SimKit interface

2.3 Simulation and Virtual Reality Interface

Real time animation of the designed system is required to get insight in to the behavior of the system during the control program execution. The VRML-based reality model of the robot assembly is a static type which cannot create animation. In our vision tool, an API interface was designed to inject scene animation into the graphical assembly in real-time. The robot graphical model is processed to organize it into a hierarchical set of assemblies with scene modifier element in each. Basic VRML language and parts are: Scenes, Nodes, and Events. 3D scenes are defined by the hierarchical structure tree of nodes (Objects). The characteristics of each node are stored in the fields. Events are used to transfer information in and out of each node.

In this visualization interface, a “Transform” node is mainly used to animate the CAD model parts. “Transform” is a grouping node in VRML format. It contains translation, rotation, scale, orientation and center fields that define the 3D information of the object. Translation field takes 3 arguments as [x, y, z] position of the object. Rotation field data type is a vector of 4 arguments [x, y, z, angle]. First three coordinates represent orientation in the 3D space and fourth element is an orientation angle, defined in radians. A Transform node has input type events as *set_translation*, *set_rotation*, *set_scale* and output type events as *get_translation*, *get_rotation*, and *get_scale*. Transform is defined in VRML format as:

```
Transform {
  exposedField SFRotation    rotation      0 0 1 3.14
  exposedField SFVec3f       translation   0 0 0
  exposedField SFVec3f       scale         1 1 1
  field        SFVec3f       bboxCenter    0 0 0
  field        SFVec3f       bboxSize      -1 -1 -1
}
```

For example, SFRotation (0, 0, 1, 3.14) defines the rotation of object around z axis with 3.14 radians value. Translation (0, 0, 0) defines the translation from origin position. Field values can be modified dynamically using an API event during the animation process. The API contains a set of methods that can be called to change the VRML scenes. Each scene is corretaled to a motion control position in the robot joints that is being produced in the simulation of the robot.

2.4 Control Program Development

A control program for pick and place motion was developed in Logix Designer. Figure 4 shows the schematics of the Denso robot and its origin coordinate system from where the pick and place motion trajectory is to be orginated. Using this information directly from the CAD model, four coordinate points P1, P2, P3, and P4 are defined in the space for pick and place application.

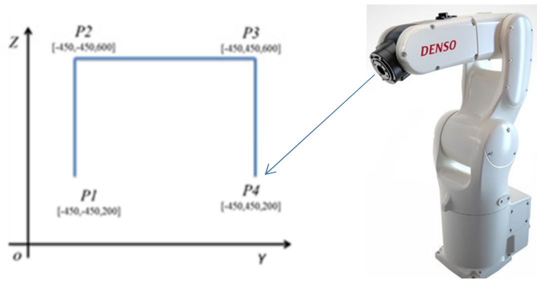


Fig. 4. Kinematics calculation

PLC motion group instructions MCT (Motion Coordinated Transform) and MCLM (Motion Coordinate Linear Move) were selected to carry out the motion control [8]. The MCT instruction is used to link two coordinate systems together. This is like a bi-directional gearing. One way to use the transform is to move a non-Cartesian robot to Cartesian positions. The MCLM instruction is used to start a single or multi-dimensional linear coordinated move for the specified axes within a Cartesian coordinate system. Then, the goal was to define the motion control algorithm and gain tunning that would be able to execute the target trajectory. The target trajectory is also shown in Figure 4. The combined effect of MCT and MCLM allowed converting the cartesian coordiantes for each target point in the trajectory into driving torque for the join actuators of the robot (join motors).

Figure 5 shows the coordinate system transform information flow that needs to happen to achieve the desired motion trajectory. MCLM instruction generates 3 axis motion profiles. In MCLM equation, the absolute or incremental move speed, maximum acceleration, and deceleration, and jerk for each axes can be specified. MCLM generates a set of 3-dimentional cartesian coordinate points from a starting point to an end point based on a desired motion speed. These coordinate points are sent to MCT to produce a target angular position. This angular position is the input to the torque generator drivers that ultimately will make the motors rotate.

MCT generates command positions for each joint actuator using inverse kinematics calculations [8] from Cartesian coordinates to joint axis coordinates. This inverse kinematics calculation is based on the dimensions and geometry of the robot that were established in the CAD model.

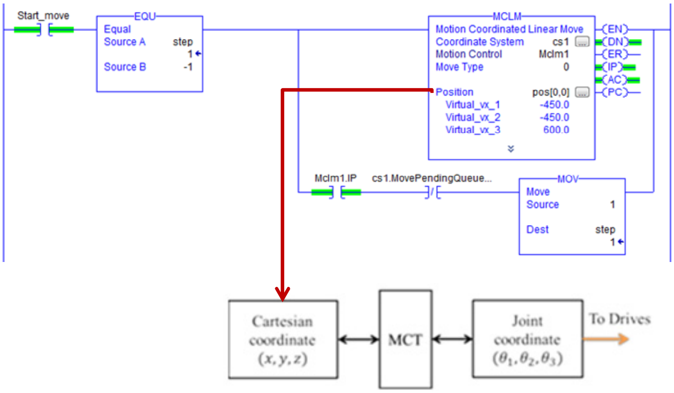


Fig. 5. MCLM instruction (b) MCT instruction

In this model, MCT produces command positions for three joints, corresponding to base, link1 and link2. Motor drives take these command position values and generate torque as an output for each joint axis movement. Current positions of the actuators are feedback to the drivers for completing the closed loop control in between the drivers and the motors. The combined effect of the different information sources helps in simplifying the programming of the PLC for this robotics application. However, this concept is not restricted to robotics and it can be applied to other systems.

3 Animation

The next step in the information flow is the integration of the motion control and system response visualization. A software application was created to generate associations (links) between graphical components and motion coordinates. The intention in this step was to remove the graphical computation burden from the simulation so to accelerate the simulation loop. In this manner, the window of observation into the simulation could be reduced to just a few seconds of simulation time. The intention is to make the simulation more pure to let it dedicate its CPU time to simulation crunching rather than graphical rendering. In motion control applications, where response times are very stringent, a feature that separates the graphical computation from core calculations can be highly valuable.

This software application creates connection tags between the CAD assemblies and the motion control tags. Based on the part and tag connection information, the simulation model is extended with a set of connecting ports (as shown in Figure 2) representing the input side signals into the simulation and the output side signals from the simulation into the visualization tool throughout the 3D API.

In the Denso robot application, there are 6 parts in the CAD assembly. These parts can be fully animated with only three motion tags effecting directly into the joint actuators. Here, the augmented CAD assemblies will take just a few input Transform coordinates to animate the whole solid model using internal coordinate and graphical calculations. These tags were connected to base, link1 and link2 parts in the assembly. This feature allows freedom to connect any part with motion commands and test it in combination with realistic simulation and motion control, as shown in Figure 6.

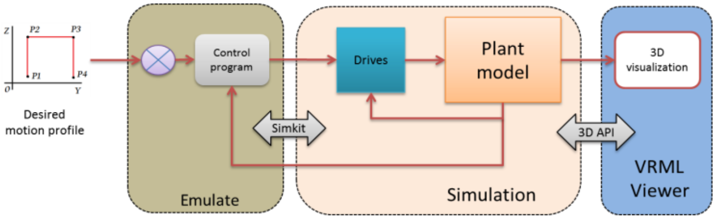


Fig. 6. SIL interface

The robot control program was downloaded to the virtual controller. Simkit linked the I/O tags of controller and simulation. Control program and plant simulation exchanged the data in pseudo real time. The robot simulation produced part rotation and translation data which were transferred to the VRML viewer throughout the 3D API. The results of the motion control algorithm that was developed for controlling this robot were checked by plotting the current position of the end effector in the simulation environment and this was compared with the commanded position.

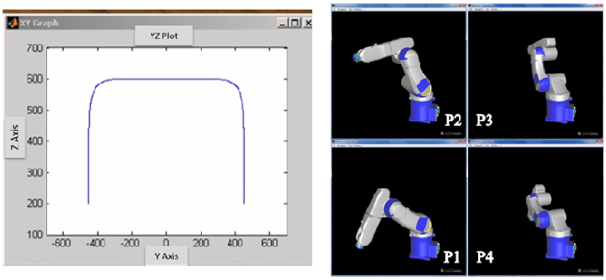


Fig. 7. Position profile response

Figure 7 shows the end effector plot and the associated position of the end effector, base and links in the 3D animator both views exhibited synchronized moves and the position profile was executed correctly by the system. Changes applied to the Coordinate values in the control program during the run time phase were immediately propagated to the simulation and visualization components.

4 Conclusion

A work flow, validation framework and process to enable a virtual design engineering environment have been presented. The environment enables interoperability of solid model design, controllers, simulation, and visualization tools into an integrated development environment. A Denso VS-6556G control application was used to demonstrate the concept. Results showed that the selected work flow and information exchange could be assembled in an automated system. The deployment of the virtual design engineering environment permitted observing in real time the effect of the controlling algorithm on the machine prototype which was simulated and controlled by motion instructions. One outcome of this works is the ability to do early validation of the controlling logic. The benefit is the reduction of engineering cost, time and resources during the design of products. Moreover, the virtual design engineering environment opens new opportunities to train engineers and operators in controls.

Acknowledgment. We would like to thank Rockwell Automation for providing the resources to carry out this work. We also would like to thank Mr. Jatin Bhatt and Mr. Bhanu Gouda from Rockwell Automation for their help in Denso robot modeling and motion control.

References

1. Boucher, M., Houlihan, D.: System Design: New Product Development for Mechatronics. Aberdeen Group (January 2008)
2. Instant reality, http://www.instantreality.org/
3. Turecskiy, A., Denery, T.: Simulation and Animation of Mechanical Systems: American Institute of Aeronautics and Astronautics
4. Schneider, D., Martin-Michiellot, S.: VRML Primer and Tutorial (1998), http://tecfa.unige.ch/guides/vrml/vrmlman/vrmlman.html
5. Wang, Q.Y., Tian, L.: A systematic approach for 3D VRML model-based assemblyin Web-based product design. International Journal of Adv. Manufacturing Technology 33, 819–836 (2007)
6. Jezernik, A., Hren, G.: A solution to integrate computer-aided design (CAD) and virtual reality (VR) databases in design and manufacturing processes. International Journal of Adv. Manufacturing Technology 22, 768–774 (2003)
7. Datasheet DENSO Robotics VS-G_Series, http://www.densorobotics.com/products/vs-g-series/data-sheet
8. Rockwell Automation, 2012 Logix5000 Controllers General Instructions Reference Manual (2012), http://literature.rockwellautomation.com/idc/groups/literature/documents/rm/1756-rm003_-en-p.pdf
9. Maturana, F., Staron, R., Carnahan, D., Iype, A., Hall, K.: Holonic Multi-Agent System for Real Time Simulation of Control Systems. In: Mařík, V., Vrba, P., Leitão, P. (eds.) HoloMAS 2011. LNCS, vol. 6867, pp. 103–113. Springer, Heidelberg (2011)

Towards an Ontology for Small Series Production

Udo Inden[1], Nikolay Mehandjiev[2], Lars Mönch[3], and Pavel Vrba[4]

[1] Cologne University of Applied Sciences
Claudiusstraße 1, 50968 Köln, Germany
udo.inden@fh-koeln.de
[2] Manchester Business School
Booth Street West, Manchester, M15 6PB, UK
nikolay.mehandjiev@manchester.ac.uk
[3] University of Hagen
Universitätsstraße 1, 58097 Hagen, Germany
Lars.Moench@fernuni-hagen.de
[4] Czech Technical University in Prague,
Technická 2, 166 27 Prague 6
pavel.vrba@fel.cvut.cz

Abstract. We describe the derivation of ontology to capture knowledge regarding small series manufacturing. This work is motivated by use-cases ranging from the assembly of large aircraft built from about 6 million parts to the production of galley inserts, both plagued by problems for arising from the same interplay between product and production engineering. During ramp-up stages of such production, there are challenges of innovative technologies, high quality and safety standards or the structural complexity of products which frequently cause supply-chain problems or require revisions of the designs that result in significant financial losses. These conditions differ significantly from standard production scenarios. Based on a detailed domain analysis, we identify five requirements for the type of knowledge to be captured in ontology, and then proceed to propose a first version of such an ontology addressing these requirements, thus providing an appropriate semantic base for intelligent, associative tools that can effectively support ramp-up management.

Keywords: Ontology, Small series production, Holonic Manufacturing.

1 Introduction

Complex product-production systems [25] like those of aircrafts are capital- and personnel-intensive ventures with series ranging from one to a few thousand of units. These small lots limit learning effects and imply frequent perturbations and changes particularly in ramp-up stages. At the same time competition drives the demands for new solutions and customization that, with factors like high safety requirements, bills of material with possibly millions of objects [21] and large work break-downs or a global supply chain accumulate to high technological risk and managerial challenges to achieve efficient utilization of labour and machinery [4], [10], [12].

Thus ramp-up phases are marked by lacks of dependability across the whole product- and production system that translate into frequent requests for changes in product or process impacting the work breakdown (Figure 1). Examples include the grounding

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 128–139, 2013.

of 50 operating Boeing 787 in January 2013 because of fire hazards from a new type of Lithium-Ion batteries which stalled production for months [23]. Airbus A380 was also delayed in 2009 because of design failures identified at ramp-up [24].

Such disturbances are a result of the interplay of a variety of factors, such as limitations of learning curves, technological complexity and dependencies between product and production engineering. We believe that agent-based support systems built on top of holonic paradigm and utilising knowledge encoded in an explicit ontological model of the product and production will help to alleviate some of these problems. In this paper we derive the domain ontology as one of the prerequisites to design and implement such systems. Whilst several ontologies are available in the literature (cf., for instance, [1], [2], [5]) not all characteristics of small series production may be taken into account. We try to reduce this gap in the present paper.

Our ontology is intended to be applied in application systems dealing with production planning and control tasks in small-series production. We believe that the decision-making abilities of commercial, state-of-the-art advanced planning and manufacturing execution systems are not appropriate. Instead, we propose a network of agent-based planners and schedulers to enhance capabilities of planning and scheduling in the target domain. The proposed ontology is a basic ingredient for the communication of the software agents. Instead of a fully automated decision-making, we understand the new system as a decision support system for human decision-makers in planning departments and on the shop floor.

Aware of the particularities of the domain and substantiated by a case-based domain analysis we shall in Section 2 analyse basic relationships between production, engineering and other domains. The conceptual details of the ontology are presented in Section 3. Section 4 discusses related literature and Section 5 presents directions of further work, particularly an extension of the model for exception-driven operations.

2 A Conceptual Analysis of the Domain

Aerospace [4] or shipbuilding [13] industries exemplify capital- and labour-intensive product-production systems. Given the complexity emerging from interdependencies of the manifold of objects, small failures can develop into large problems, occurrence made more likely by the scale of complexity but also by the nature of production. Indeed, measured by the bill of materials the use-cases that motivate our work differ by a factor of 10000, yet firms on either ends face complexity emerging from the same sources.

An example: A part showing a failure that may have slipped through tests now needs the decision of production management, quality management and engineering whether, and if, when and under which conditions work can be continued. If contingency buffers are exhausted the failure will propagate – the more the longer it takes to solve it. But then engineering may induce more impact if, e.g., technology is changed, previous assemblies to be revised (rework), or technicians required to obtain new skills. Finally the problem may abridge the order book and damage reputation.

The analysis starts from the concurrent effort of product and production engineers (Figure 1) to specify technology (left side) and its breakdown into processes (right) [7]. It suggest, that in contrast, e.g., to the PROSA model that considers product-, resource- and order-knowledge to be the conceptual backbones of holonic production systems [16], the domain is organised by engineering and process knowledge: Implementing a valid technology also requires the specification of a breakdown of work in terms of as a disambiguated operations network integrating processes, materials or parts, technical resources, infrastructures, and human actors as well as suppliers and related audits, tests or approvals. In aviation industry, this network is obligatory and is a subject of legally enforced proceedings and personal accountability [26]. The managerial rationale deriving from these factors can be described as *sustain certified processes or at least mitigate impact of change – both without compromising quality.* This translates into **requirements** to ontological models of the domain:

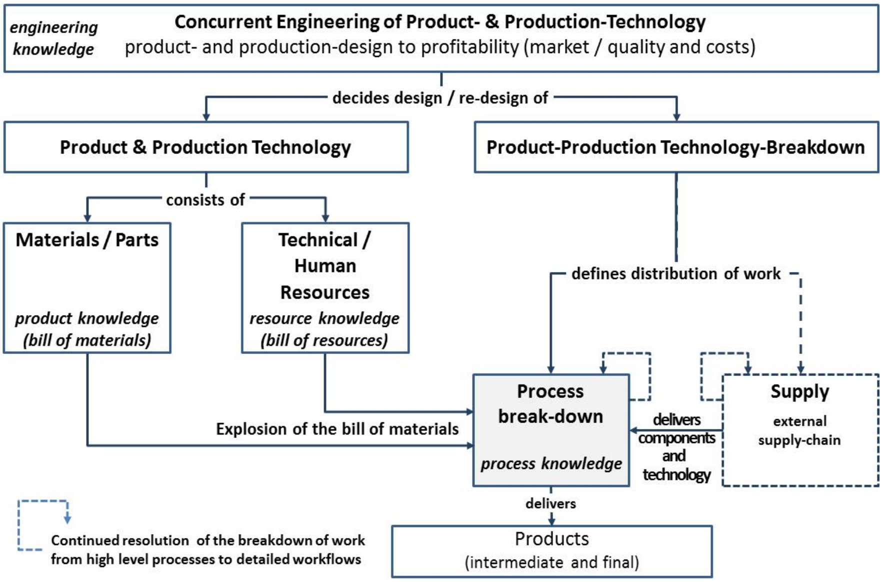


Fig. 1. The Structure of Knowledge and the Breakdown of Work and Objects

(R1) Models should reflect interactions between the product and its production, providing support for ripple effects when changing both during ramp-up.
(R2) Due to frequent change the ontology needs to allow keeping track of design rationale for specific product or process engineering choices, and the actors responsible for making these decisions and those responsible for implementation.
(R3) Models should include the interactions between engineering, the administration and the logistic sub-system, because of their important roles in ramp-up.
(R4) To offer a maximum of flexibility, the model should record true dependencies between processing steps, e.g., re-scheduling without ordering dependency, or re-allocating resources to steps where they are of a higher value. Though limited at process level, flexibility will be present on levels of detailed operations.
(R5) The core structure of the model should apply to different stages and aspects of manufacturing – engineering, production, and logistics.

Figure 2 depicts a high-level structure of the domain where the technology enfolded in the product-production system, the engineering departments that design and redesign this technology, the logistics and departments required to dynamically allocate inventories and the administration (business administration) form a multidisciplinary network coordinating their response to frequent unplanned events.

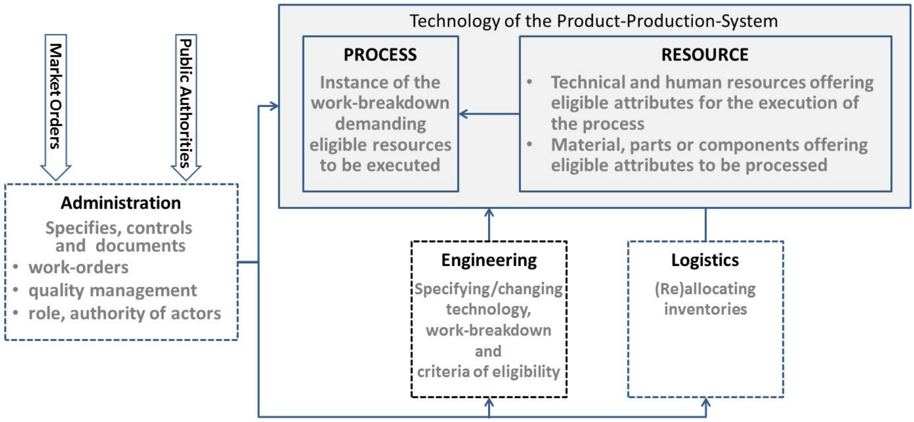


Fig. 2. An Extended high-level Domain-Model of a Product-Production System

(1) **ADMINISTRATION** is to be certified pursuant to EASA regulations 21G (producer) and 21J (developer) of aerodynes [26]. Administration particularly includes QUALITIY MANAGEMENT regarding the compliance of work and documentation to rules, and the airworthiness of resources and products. Further units contract or estimate market orders (marketing, sales, customer services) and others translate this into WORK-ORDERS with goals and control parameters (target budgets of consumption, quality measures, date of delivery …) after processing internal objectives and plans and control the financial efficiency. The administration is responsible for and defines ROLES and responsibilities of actors.

(2) **PROCESS:** The work-breakdown creates a class of well defined processes, each delivering an intended output and together the final product. Each process is started and instructed by a work-order. The resolution of detail depends on requirements to control processes and may go down to individual activities performed by individual resources executing an order.

(3) **RESOURCE** is the class of objects executing activities in response to orders [7]. Resources need to offer a profile of attributes that is demanded by a process, specified by the product-production technology in terms of skills or functionality. On high level and in the set-up of mandatory work-breakdowns also constituents of the PRODUCT (e.g. parts) can be conceived as resource to the process.

(4) **LOGISTICS** has the task to take *the right objects at the right time to right place*, i.e., to (re)allocate LOGISTICS OBJECTS (particularly production resources) to demand in space, time and administration [20]. Under fast-changing ramp-up conditions the intelligence of logistics services may contribute to the mitigation of problems. Therefore logistics is controlled by orders and obliged to meet requirements specified by engineering, e.g. for storing or transporting objects.

(5) **(Re-)ENGINEERING** of product-production networks results from events like late customer requests (e.g., for further customization) or non-conformity of parts or a technological improvement. There are a few papers about ontologies on engineering [17], [18], [19], but here we focus on interactions between engineering and production, e.g. the duration of interruption of production until a solution is ready to be implemented, the extent this solution changes the product respectively the production technology, or the need to train staff to implement the solution.

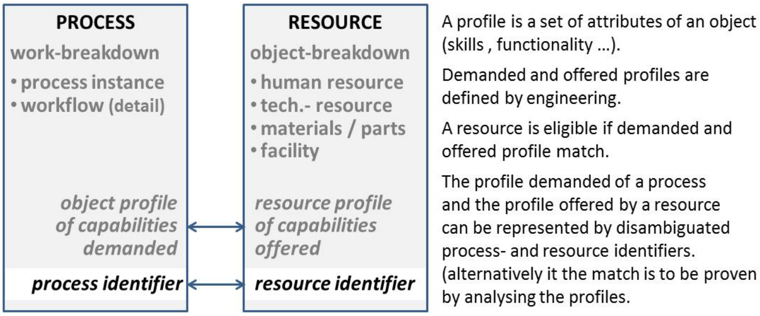


Fig. 3. Eligibility – the Relation between Process and Resource

Figure 3 shows a second high-level concept of fundamental interactions between the classes of processes and resources. This has been determined in the engineering of

DOMAIN	Substructures	Process	Resource

Administration	• Department
	• Decision (Output)
	  • Work-order
	  • Target agreement
	  • Design Query
	• Document (Output)
		• Decision making
		• Controlling
		  • Quality
		  (process, product, resource)
		  • Financial efficiency
		• Documenting
			• Actor (autonomous)
			  • Decision maker
			  • Assistant
			  • 3rd Party Employee

Production	• Manufacturing
	  • Part (Output)
	• Assembly
	  • Component (Output)
	  • Module (Output)
	  • System (Output)
	  • Final Product (Output)
		• Goods acceptance
		• Work-preparation
		• Execution
		• Testing
		• Rework
		  • Re-assembly
		  • Disassembly
			• Actor (autonomous)
			  • worker / technician
			  • supervisor …
			• Product constituent
			• Facility
			  • Production-line
			  • Machine, Tool, Jig , Crane …
			• Implement
			• Expendable (secondary resource)

Logistics	• Stationary Inventory • Underway Inventory • Output Inventory	• Inventory control • Pick / kit / pack • Transport • Delivery • Return of shipment	• Actor (autonomous) • Storage • Container • Pallet • Vehicle

Engineering	• Technology (Output)
	• Breakdown of work
		• Requirements definition
		• (Solution finding*)
		• Testing & validation
		• Implementation
			• Actor (autonomous)
			  • engineer
			  • accountable officer …
			• Systems (CAD, Testing …)

	Attributes		
	Identifiers	Constraints	Objectives (Actors)

	• disambiguated ID
	• Affiliation to
	  • organisation ID
	  • process ID
	  • work-order ID
	  (with reference to batch or individual market-order)
		• Profile
		(skills, functionality, capacity)
		• Availability …
		(where, when, how long …)
		• Dependency
		• Endowment
		(budget, consumption)
			• Individual goals
			(within scope of autonomy)
			• Control parameters
			  • by Work-order
			  • by Target agreement

Fig. 4. Exemplary Substructures of Domains

the work-breakdown (Figure 1, [7]): Each process requires from resources a particular profile (e.g. a particular type and precision of a wrench with torque control).[1]

High-tech industries in general invest significant efforts into the implementation of technological standards or the disambiguation of identifiers of processes, resources, or of actors. These identifiers can be used to operationalize as well as reason about the fit between capabilities asked by processes and offered by resources.

Figure 4 gives an overview of further details: major processes, substructures and attributes. The first column lists the operations' domains introduced above with *production* (the operating product-production system) and depending on operations in *logistics* and *engineering,* together governed by the *administration*.

Each domain includes substructures and delivers an *output.* In *production* this is an *intermediate* or *final product* (cf. e.g. [2]). The *administration* delivers *decisions* (plans, contracts, work-orders, budgets), or *documents* related to commercial or legal reasons. *Logistics* 'produces' *inventories* hold in a warehouse or being on the road, expected to arrive at a point of time. Finally *technology* and the breakdown of work and related product and production technology are outputs of *engineering*.

Processes depend on technologies and decisions on work breakdowns or on allocations of authority. In engineering *solution finding*, e.g., responding to failure includes contributions of human creativity, that is hard to capture in any conventional model.

Resources consume *expendables* (e.g. electric power). This consumption may be subject of a *budget* and, with the goal to improve the economic performance, and comparing planned and actual consumption may induce action to reduce idleness of resources, or the consumption of environmental certificates. The idea of *implement* refers to auxiliary resources serving in response to non-planned events, like workspace that is only to be planned if it is occupied by unfinished preceding work.

Also major **attributes** have been identified, enabling both (Figure 4), the conformity to technological and administrational frameworks and the interoperability between the domains: The group of *identifiers* ensures an disambiguated identity of an object and its affiliation to an administration (e.g., an object is owned by X) its assignment to a process (continuous, current), the work-order controlling it or and related market contracts. *Constraints* refer to side-conditions to choices of flexibility in operations, e.g. a *profile* of *skills* or *functionality* and *capacity* of resources that may qualify to serve in one specific process only. *Availability* is the readiness of a resource to execute a work-order depending on its *place* in a point of *time*, the *duration* of a job or on its *affiliation*. In the reach of *autonomy, objectives* drive the behaviour of actors again controlled by the content of the work-order or by a target agreement.

3 Conceptual Model of the Proposed Ontology

We follow the definition viewing ontology as an explicit specification of a conceptualization of objects and their relations that exist in a particular domain [3]. In the context of this paper we use ontology to capture knowledge about the domain of ramp-up manufacturing and to encapsulate this knowledge into a form suitable for agent reasoning and providing support for ramp-up processes in this domain.

[1] Similar rules also apply to rail (European Railway Agency) or to car industry [27]. Even without enforcement competition and operations' complexity coerce to ensure control of technical quality [25].

To create the ontology presented as an abstract UML class model in Figure 5, we have built upon a number of existing enterprise and manufacturing ontologies, including the Enterprise Ontology [14], but also [9], [10]. This ontology captures an initial conceptual level of thinking about the domain, and is yet to be enriched with classes and relationships corresponding to design-style considerations such as how to avoid millions of "part" and "resource" subclasses in an Airbus 350 ontology. These have not been introduced to help retain conceptual clarity of the overall knowledge schema.

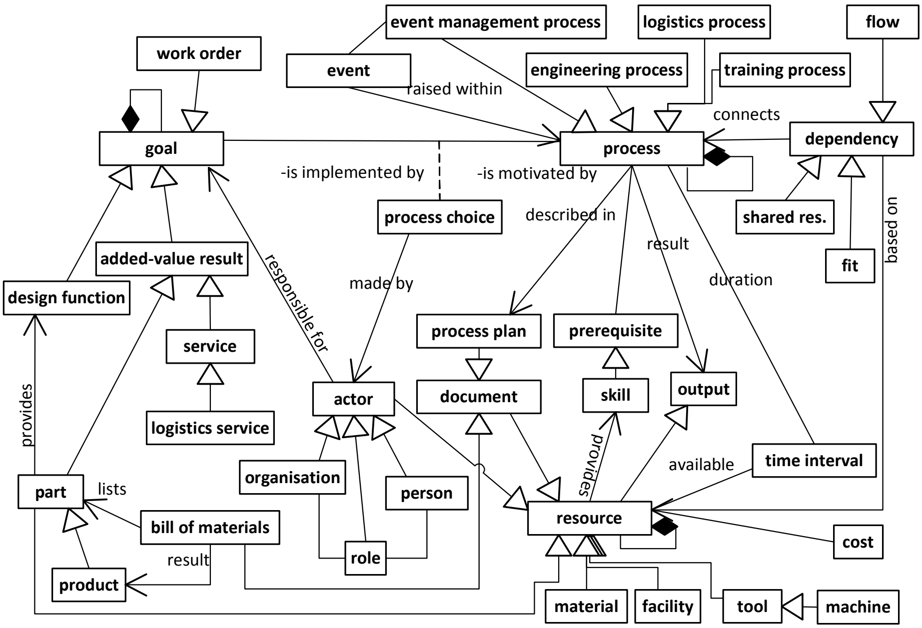


Fig. 5. UML Class Model of the core Small Series Manufacturing Ontology

To reflect the *Requirement R1* for the interplay between product and production engineering, we position in the centre of our ontology the *processes* and their motivating *goals*. The goal of manufacturing is to bring parts into existence respectively to assemble larger ones (e.g., a wing or an airplane) from smaller ones. In production we often deal with creation of a part or a set of *parts* according to a work order. This can be decomposed into the creation of a number of subparts which are then assembled into the part according to *bill of materials*.

Goals are also encountered in other ontologies, for example they feature in the Enterprise Ontology [14] as a subclass of "Purpose". However in our ontology goals are not roles in relationships as in the Enterprise Ontology, which allows us to link them directly with entities from the domain, to allocate them as responsibilities, and to link them to the processes chosen to achieve each goal so that we can retrieve the original goal if a process fails, and to select an alternative way of achieving the goal (if possible). Goals are generalisation of a wide range of domain terms, from design objectives to work orders.

Processes are decomposed into steps (also referred to simply as processes because of the recursive decomposition). Processes can specialise other processes (making an airplane as specialisation of making a vehicle).

Reflecting *Requirement R2*, we introduce the concept of *an actor*, which is a generalisation of *administration*, *person* and *role*. An actor is responsible for a goal and can choose to either decompose the goal into sub-goals which can be delegated, or to implement this goal using a process. Often these choices will be limited by technology and process constraints to ensure quality of production as designed by the engineering department, still the details of certain low-level operation are left at the discretion of the responsible actors, for example “connect part A to part B” (or “paint wing”). A person acts in a role assigned by the administration (quality controller, station manager). The concept of role often implies a set of rights and obligations.

The link between *actor*, *goal* and *process* allows us to effectively revisit and revise design decisions about both the product and the process, when failures are raised at ramp-up time, or when change requests come in from customers.

The decomposition and delegation of goals and processes allows us to implement *Requirement R3* by modelling the interactions between engineering goals (design a high-lift wing), production goals (assemble wing) and logistic goals (provide wing assembly components), together with the administrational units responsible for them (*engineering*, *production* and *logistics departments, instances of administration*).

To reflect *Requirement R4*, we introduce elements of Coordination Theory [6], where the conventional process specification using sequence of process steps (or activities) is avoided, and three types of *dependencies* between process steps using the inherent role of *resources* are used instead. These three types of dependencies are *shared resource* (the same resource is used at two places), *flow* (the output of one step is used as a prerequisite for another), and *fit* (two steps should happen at the same time since they result in the same resource, or a part. This is one of the innovative elements of our ontology which distinguishes it from other manufacturing ontologies and only allows us to exploit the maximum possible flexibility when re-planning to minimise the effects of disruption.

Parts and *actors* are *resources* needed for *processes*, and establish dependencies between process steps. A process is linked to resources via two properties: It usually produces an *output*, which may be a physical resource such as a part or a conceptual resource such as *skill* (resulting in additional skills) or a *service*. A process also has *prerequisites*: resources needed for the process to run. Some will be consumed and may need to be provided like a budget or also the content of the tank of a lorry. Others may improve (human skills) [22], or remain unchanged (functionality of tools).

Examples of resources include raw material, parts, actors and *tools* (simple ones like screwdrivers or complex ones such as robot *machines*). Other types of resources include *facilities* (room/space), *documents* (drawings, instructions, formal documentation), and *materials* (steel, plastic, etc.). A resource also has availability and cost. Monitoring for the presence of all prerequisites is done by *event management* processes, which can raise and process *events* associated with a particular process, e.g. if the process has not delivered output satisfying a standard, or if not all prerequisites are available (worker is off sick, or parts are not supplied in time).

To reflect the final *Requirement R5*, we have structured the core ontology in a way which applies to design, production, logistics, maintenance and even training, as all these sub-domains of manufacturing contain actors who pursue goals by either delegating these goals to other actors, or by following processes which bring about the achievement of these goals. Goal is thus defined in a suitably wide manner, for example the concept of goal is specialised into two main types: creation of value (producing a product or delivery of a service), and achievement of function (wings are a design choice to provide an uplift function). The delivery of a service includes a delivery of logistics services, which change the location of resources and parts.

Populating the ontology with knowledge happens gradually throughout the concurrent design of product and process, creating a self-documenting artefact about the complex links existing between design (engineering), production (assembly) and logistics (supply chain). This self-documentation property allows agent reasoning to identify the consequences of a process or a product fault, and to re-plan process steps thus minimising the delay caused by the fault (or a change request).

4 Related Work

Ontologies discussed in the literature for the manufacturing domain can be organized into two major groups: general domain ontologies and specific-purpose ontologies. The first type is usually created as a part of holonic/agent-based production control systems to support knowledge sharing and reasoning while specific-purpose ontologies are developed for a particular scenario, mostly assembly, and then integrated in agents controlling processes. The idea is always the same: describe manufacturing resources and operations they can provide and use the ontology-based description of production processes to automatically execute a series of operations [16].

An integration of ontology in agent-based assembly line control systems is presented in [29] defining two basic categories of concepts: *modules* and *skills*. Modules represent physical manufacturing resources or their aggregation, like a workcell, workstation, or processing unit. Skills are capabilities to perform manufacturing actions, as for instance MoveLinear, which are provided by the modules. A similar application of ontology for assembly domain is presented in [31]. The basic concepts are *material resource* and *operation* with the same meaning as modules and skills from the previous example. To express the fact that a particular machine provides a particular operation the relationship *enables realisation of* can be used to link resource and operation. Another application can be found in [30] structuring the ontology in three layers: (i) *product* presented as a hierarchy of subassemblies and parts, (ii) *activity* representing basic action that changes the state of the product including product order, work order and task, and (iii) *resource* representing a physical component able to perform a certain action.

An ontology targeting discrete production processes controlled by agents is presented in [16]. In a simplistic form, it describes orders, i.e. what features the product should have, the production process, i.e. an ordered sequence of steps, each of which contains required resources including raw materials or sub-assemblies, the configuration of assembly stations, and finally details of the factory transportation system.

The ontology of the ADACOR architecture discussed in [1], [2] was designed for representing the assembly processes. However, it is rather generic and does not contain enough details for complex assembly processes. Another ontology of assembly processes is presented in [5]. The ontology is an extension of the National Institute of Standards and Technology's Process Specification Language (PSL). The product is defined as a hierarchy of assemblies, components and parts linked together by liaisons. It defines how parts and components of a product relate to each other. While this ontology is appropriate for modelling assembly, it neither includes resources nor provides a description of any disturbances typical for small-series production.

Specific ontologies for complex job shops motivated by the semiconductor wafer fabrication facilities are proposed in [6]. However, in the setting huge process plans including a large number of operations with precedence constraints are typical, but the bill of materials (BOM) is rather simple. The FRISCO ontology designed to support organization of knowledge in automotive supply chains is reported in [28]. It contains a set of models for description of various aspects like products, customers, suppliers, manufacturing, resources, planning and scheduling, etc. The OZONE ontology [11] is intended to support scheduling systems. It offers various types of resources and constraints that are typical for scheduling problems. However, again assembly-specific concepts are not addressed by this ontology.

5 Conclusions and Further Work

Our work focuses on manufacturing scenarios marked by a high undependability of operations. The objective is to demonstrate capabilities of semantic technologies for planning and control that enable associative capabilities to flexibly explore and evaluate options to mitigate (to the extent possible) unpredictable interruptions. Based on the analysis of industrial use-cases, the current high level model identifies core concepts and integrates them in a coherent model in order to meet these goals. In order to support the development of strategies of interoperability between production, logistics as well as engineering, the model is designed to be applicable to each of these domains without compromising governing technological and economic rationales. The analytic part delivers the substructures and major attributes of these domains that substantiate the projection of an initial conceptual semantic model to be elaborated, extended and validated in future work.

Further steps of the development of the ontology will see details about the concepts of time, costs and orders added in, and an ontology of events: Considering the large variety of possible disturbances and the context-sensitivity of failure propagation will be in the core of our future work on the ontology of events, aiming to formalise policies for handling unplanned events. In order to enable support for appropriate decisions, we will have to integrate the concepts of time, costs, and risk, including eventually the "time to amortization" concept pertinent to large investments.

We are planning to test the ontology in three stages, starting with an informal application on two case studies (aircraft and galley inserts). The ontology will be updated with lessons learned from these applications, and, using Protégé, transcribed in OWL. This will provide opportunities to consider lower levels of detail such as cardinality of relationships, ranges and domains of properties, etc. In the third stage, we will develop agent-based prototypes which would use the ontology, and our attempts at initial applications of the prototypes will allow further elaborations of the ontology.

Acknowledgement. This work was supported in part by the European Union FP7 Programme under the project ARUM, Grant Agreement No. 31405.

References

1. Borgo, S., Leitão, P.: The role of foundational ontologies in manufacturing domain applications. In: Meersman, R., Tari, Z. (eds.) CoopIS/DOA/ODBASE 2004. LNCS, vol. 3290, pp. 670–688. Springer, Heidelberg (2004)
2. Borgo, S., Leitao, P.: Foundations for a Core Ontology of Manufacturing. In: Sharman, R., Kishore, R., Ramesh, R. (eds.) Ontologies: A Handbook of Principles, Concepts and Applications in Information Systems. Integrated Series in Information Systems, vol. 14, pp. 751–776 (2007)
3. Gruber, T.R.: Towards Principles for the Design of Ontologies Used for Knowledge Sharing. Int. J. Hum. Comp. Studies 43(5/6), 907–928 (1995)
4. Heike, G., Ramulu, M., Sorenson, E., Shanahan, P., Moinzadeh, K.: Mixed Model Assembly Alternatives for Low-volume Manufacturing: The Case of the Aerospace Industry. Int. J. Production Economics 72, 103–120 (2001)
5. Lose, N., Hirani, H., Ratchev, S., Turitto, M.: An Ontology for the Definition and Validation of Assembly Processes for Evolvable Assembly Systems. In: Proceedings of the 6th IEEE International Symposium on Assembly and Task Planning: From Nano to Macro Assembly and Manufacturing, pp. 242–247 (2005)
6. Malone, T.W., Crowston, K.: The Interdisciplinary Study of Coordination. ACM Computing Surveys 26(1), 87–119 (1994)
7. Mas, F., Rios, J., Menendez, J.L., Gomez, A.: A Process-oriented Approach to Modeling the Conceptual design of Aircraft Assembly Lines. International Journal of Advanced Manufacturing Technology 67(1-4), 771–784 (2012)
8. Mehandjiev, N.D., Stalker, I.D., Carpenter, M.R.: Recursive Construction and Evolution of Collaborative Business Processes. In: Ardagna, D., Mecella, M., Yang, J. (eds.) BPM 2008 Workshops. LNBIP, vol. 17, pp. 573–584. Springer, Heidelberg (2009)
9. Mönch, L., Zimmermann, J.: An Ontology to Support Adaptive Agents for Complex Manufacturing Systems. In: Proceedings 3rd IEEE International Conference on Computer Software and Applications, pp. 531–536 (2008)
10. Noack, D., Rose, O.: A Simulation Based Optimization Algorithm for Slack Reduction and Workforce Scheduling. In: Proceedings of the 2008 Winter Simulation Conference, pp. 1989–1994 (2008)
11. Obitko, M., Mařík, V.: Adding OWL Semantics to Ontologies Used in Multi-agent Systems for Manufacturing. In: Mařík, V., McFarlane, D.C., Valckenaers, P. (eds.) HoloMAS 2003. LNCS (LNAI), vol. 2744, pp. 189–200. Springer, Heidelberg (2003)
12. Rios, J., Mas, F., Menendez, J.L.: Aircraft Final Assembly Line Balancing and Workload Smoothing: A Methodical Analysis. Key Engineering Materials 502, 19–24 (2012)
13. Storch, R.L., Lim, S.: Improving Flow to Achieve Lean Manufacturing in Shipbuilding. Production Planning & Control 10(2), 127–137 (1999)
14. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The Enterprise Ontology. The Knowledge Engineering Review 13(1), 31–89 (1998)
15. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference Architecture for Holonic Manufacturing Systems: PROSA. Computers in Industry 37(3), 225–276 (1998)

16. Vrba, P., Radakovič, M., Obitko, M., Mařík, V.: Semantic Technologies: Latest Advances in Agent-based Manufacturing Control Systems. Int. J. of Production Research 49(5), 1483–1496 (2011)
17. Tudorache, T.: Employing Ontologies for an Improved Development Process in Collaborative Engineering. Dissertation, Fakultät IV – Elektrotechnik und Informatik der Technischen Universität Berlin (2006), http://opus.kobv.de/tuberlin/volltexte/2006/1437/pdf/tudorache_tania.pdf, Reference: http://protegewiki.stanford.edu/wiki/Engineering_ontologies (accessed September 12, 2012)
18. Gruber, T.R., Olsen, G.R.: An Ontology for Engineering Mathematics. In: Doyle, J., Torasso, P., Sandewall, E. (eds.) Fourth International Conference on Principles of Knowledge Representation and Reasoning (1994)
19. Siang, K.D., Duffy, A.H.B.: Towards an ontology of generic engineering design activities. Research in Engineering Design 14(4), 200–223 (2003)
20. Arnold, D., Isermann, H., Kuhn, A., Tempelmeier, H., Furmans, K. (Hrsg.): Handbuch Logistik 3. Auflage. Springer, Heidelberg (2008)
21. 747 Fun Facts, http://www.boeing.com/boeing/commercial/747family/pf/pf_facts.page (accessed November 2, 2011)
22. Terwiesch, C., Bohn, R.E.: Learning and process improvement during production ramp-up. International Journal of Production Economics 70(1), 1–19 (2001)
23. Dreamliner Grounded Until At Least End Of May. Sky News (February 25, 2013), http://news.sky.com/story/1056514/dreamliner-grounded-until-at-least-end-of-may (accessed March 27, 2013)
24. Kingsley, J.M.: Airbus slows A380 final assembly ramp-up. Flightglobal (May 14, 2009), http://www.flightglobal.com/news/articles/airbus-slows-a380-final-assembly-ramp-up-326416/ (access February 25, 2013)
25. Takeishi, A., Fujimoto, T.: Modularisation in the auto industry: interlinked multiple hierarchies of product, production, and supplier Systems. Int. J. of Automotive Technology and Management 1(4), 379–396 (2001)
26. European Aviation Safety Agency (EASA), accessed September 30, 2012: Commission Regulation (EC) No 1702/2003, September 2003: Implementing Rules of Airworthiness, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CONSLEG:2003R1702:20091228:EN:PDF
27. Automotive Regulatory framework for type-approvals, http://ec.europa.eu/enterprise/sectors/automotive/technical-harmonisation/regulatory-framework/ (accessed March 12, 2013)
28. Hellingrath, B., Witthaut, M., Böhle, C., Brügger, S.: An Organizational Knowledge Ontology for Automotive Supply Chains. In: Mařík, V., Strasser, T., Zoitl, A. (eds.) HoloMAS 2009. LNCS, vol. 5696, pp. 37–46. Springer, Heidelberg (2009)
29. Câmdido, G., Barata, J.: A Multiagent Control System for Shop Floor Assembly. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 293–302. Springer, Heidelberg (2007)
30. Merdan, M.: Knowledge-based multi-agent architecture applied in the assembly domain. Dissertation thesis (PhD). Vienna University of Technology (2009)
31. Al-Safi, Y., Vyatkin, V.: An Ontology-Based Reconfiguration Agent for Intelligent Mechatronic Systems. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 114–126. Springer, Heidelberg (2007)

Ontological Extension of PROSA for Manufacturing Network Formation

Guiovanni D. Jules, Mozafar Saadat, and Salman Saeidlou

School of Mechanical Engineering, University of Birmingham, United Kingdom
{gdj039,m.saadat,sxs989}@bham.ac.uk

Abstract. The scope of the research is the inter-enterprise communication which is required for the formation of manufacturing network. The overall aim of the research is to develop a simulation platform for the formation of manufacturing networks, scheduling of jobs in the networks and an intelligent assistance system for mending the networks in the event of disturbances. Ontology is normally built from a foundational ontology. But in this paper, we use a reference architecture. The Product-Resource-Order-Staff Architecture (PROSA) is used as the basis for ontology development. This paper presents the development of an ontology used in the formation of manufacturing networks. The ontology is developed to support the entities, their interaction protocol and the knowledge emerging during the auctioning process in the manufacturing domain. The ontology is also used to enable an illustrated investigation of the interactions during the auctioning process. The methodologies involve ontology editing, object-oriented programming and the visualization of the objects and interactions during the network formation and scheduling process.

Keywords: Ontology, manufacturing network, planning, scheduling, PROSA, contract net protocol.

1 Introduction

Globalization has produced both winners and losers. The winners have been large multi-national companies while small businesses have struggled. As of 2010, small enterprises have accounted for 99.3% of UK enterprises, generated 48.2% of UK employment and contributed 35.7% of total sale revenues to the UK economy. This constitutes 7.755 million jobs being sustained by small enterprises [1]. It is a reality that the survival of small businesses is down to outsourcing and frequent re-configuration of outsourcing strategies, tactics and operations [2]. A survey revealed that in Q3 of 2012, 53% of manufacturers worldwide were outsourcing some non-core competencies. Reducing costs, improving customer focus, accelerating company re-organization, gaining access to management expertise unavailable internally are the primary motivation for businesses to outsource [3]. More specifically working in networks becomes an essential survival strategy. By participating in manufacturing networks, the manufacturers can engage with a project managing company that attracts the business and whose customers' needs are its primary focus. In this way,

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 140–151, 2013.

business opportunities can substantially increase by attracting new customers and retaining the existing ones.

2 Background

In this section, the term manufacturing network is elaborated, an industrial case study is presented and the details of manufacturing network formation are explained. Next, the manufacturing ontology, as the state-of-art for knowledge representation in the manufacturing domain, is presented.

2.1 Manufacturing Network and Industrial Case Study

A manufacturing network is the temporary integration of manufacturers and the project managing company. The manufacturing network is formed from a pool of eligible manufacturers known as a collaborative networked organization (CNO) or virtual breeding environment [4]. When bidding for orders, the manufacturing networks compete whereas individual manufacturers cooperate. Manufacturers which would normally be competing now collaborate with each other to make the network as a whole more competitive. This is called co-opetition [5].

GFM Srl is a project managing company to a number of manufacturing networks during production. The unconventional Italian company manages a group of small and medium manufacturers primarily based in the region of Bergamo, manufacturing parts for the power generation industry. The company fits a theoretical model of manufacturing which was coined 18 years ago as part of the Intelligent Manufacturing System initiatives [6]. The planning and scheduling of manufacturing network for one-of-a-kind production is the interest of this paper and the motivation is to find ways to fulfil the potential of collaborative network organisations in that field of research. Manufacturing network is a fairly new tactical approach and an advanced IT infrastructure is instrumental for integration and seamless coordination [7]. The full potential of manufacturing networks in terms of mass customisation, one-of-a-kind production types and more collaborative manufacturing has not been seen yet. The new context of manufacturing networks is the motivation behind the use of emerging IT enablers to improve scheduling and planning.

2.2 Ontology in Manufacturing

Ontology has been used to model the manufacturing resources of a shop floor producing electronic connectors [8]. However, the scope of ontology presented did not encompass operational and control information relevant to the production such as resource schedules and the resource performance is not modelled. Another study using ontology modelled the logistics, technical and control aspect of products [9]. However the fundamental concepts used were not clearly identifiable so that when the ontology is scaled out, its maintenance may become problematic. A further criticism of this study is the problem caused by the lack of semantic provision for representing historical data such as order tardiness and quality defects is a problem. Ontology designed to model the semantics of collaborative manufacturing is clearly divided into that relating to customer, product, manufacturer or transport and is consequently easily maintained, however it is semantically incompatible for representing shop floor production or production in manufacturing network [10]. Critically the ontology pertaining to this study is well designed and the description of a manufacturing plant is accurate [11]. The ontology uses the holonic manufacturing concept. MASON is one of the most comprehensive ontologies in the manufacturing context and one of the most accessible ontologies. Its aim is to create a common upper ontology of the manufacturing domain and is founded on the concepts of entities, operations, and resources. As such it was developed to enable the automatic cost estimation of products [12]. PABADIS manufacturing ontology is one of the best ontologies to capture the conceptual information about each resource, each product and each operation in the manufacturing domain [13]. The objective of PABADIS is focused towards optimising the product performance. The ADACOR manufacturing ontology is an ontology that is defined by concepts from the holonic abstraction of the manufacturing domain [14]. It is a strong tool for the control of the manufacturing system on the shop floor. In the context of manufacturing networks, the PABADIS and ADACOR manufacturing ontologies need to be adapted to account for aggregates of manufacturing resources i.e. a manufacturing network is made of many manufacturers. Also the ontologies provide a strong data model and metadata model for the control manufacturing system. The PABADIS and ADACOR projects present an integrated approach to planning, scheduling and control which is reflected in their respective ontology. However, an ontology for the planning and scheduling of manufacturing is not explicitly demonstrated. The planning and scheduling is the result of the interaction of entities in the manufacturing domain and the interaction can be modelled as an auctioning process. The communication pattern involved in the auctioning process is modelled by the Contract Net Protocol (CNP) in steps: call for proposal (announcement), bidding (proposal), awarding, and informing [15]. An ontology that supports this process is not explicit in the PABADIS and ADACOR manufacturing ontology.

3 Methodology

Representation of ontology for the manufacturing network planning process using graphs is not prominent in manufacturing research literature. Much of the literature on domain specific ontologies has been represented as XML snippets. Graph is a great way of communicating clearly what goes on in a process such as network planning. Ontoviz is a highly sophisticated graph visualisation tool that is plugged into the ontology editor Protégé to visualise ontologies. Therefore, graphically, this paper demonstrates objects, relationships, information, and the sequence of actions involved in the planning of manufacturing networks.

3.1 Ontology Development

There exists many ways of defining an ontology. The structure of the ontology is based on the choice of the concepts and defining the structure is an iterative process.

The first step in developing an ontology is to identify the domain of expertise that needs to be represented. The second step is to agree on the role that the ontology will fulfil. The third step is to establish what entities will interact with the ontology and what information will be queried from them. The fourth step is to find existing and reusable ontology to build on. Else, the ontology is built, guided by a reference architecture if available. The reference architecture helps to define concepts and relations between them. In the fifth step, the properties of the concepts such as the value restrictions and the cardinalities are expressed. And finally, the competency of the ontology is evaluated against the domain, purpose and query efficacy that it was specified to achieve [16].

3.2 PROSA Inspired-Ontology for Manufacturing Network Concept Modelling

PROSA stands for product, resource, order, staff architecture and introduces the concept of basic holons namely product holons, resource holons, order holons and staff holons [7]. The three basic holons and their aggregates and specialisations can cover all critical manufacturing functions to construct a manufacturing system [17]. Fig. 1 shows the simplified graph of ontology for modelling manufacturing networks.

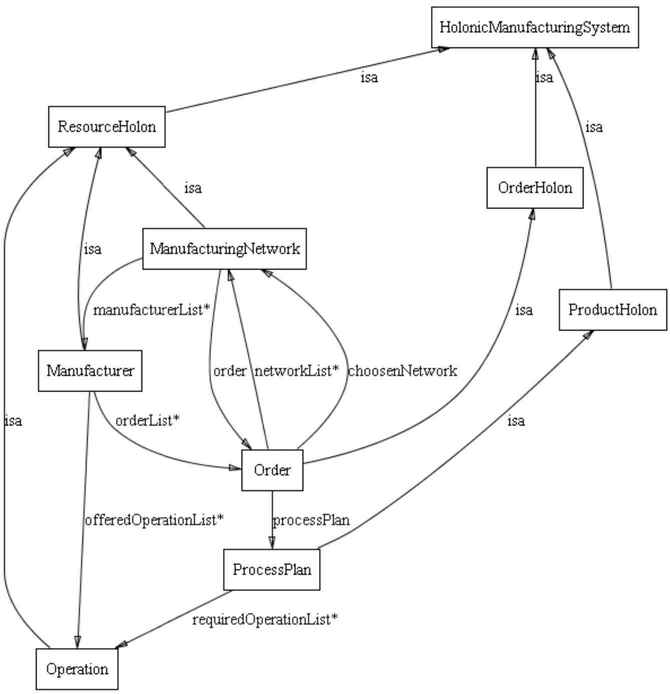


Fig. 1. A 3-level ontology for the system of entities involved in a manufacturing network

Process planning is the typical function carried out by the product holon HMS:ProductHolon and short term job scheduling is performed by the order holon HMS:OrderHolon. In addition, factories, networks of factories and manufacturing operations are production means represented by the resource holon HMS:ResourceHolon.

Process knowledge describes the offered capabilities of the resource and the required capabilities by the order. This involves operation descriptions such as ResourceHolon:Operation and ProcessPlan:Operation. Production knowledge relates to the possible sequences of processes ProductHolon:ProcessPlan to be executed on the resources ResourceHolon:ManufacturingNetwork and ResourceHolon:Manufacturer. Process execution knowledge is concerned with the control of processes on the resources and monitoring of the progress of execution.

3.3 Network Elements Modelling

In order to build the model of the manufacturing network within the scope of planning and scheduling, the objects involved in those activities must be modelled. The scope of network modelling requires manufacturers to be considered at the macro-level only instead of the micro-level of machines.

The manufacturer is modelled as the subject ResourceHolon:Manufacturer and the predicates are defined in Table 1. The action to be taken on the objects is implemented as predicate bids.

Table 1. Predicates of subject ResourceHolon : Manufacturer

is_named	has_operation_scheduled_at	offers_operations
has_bids	made_successful_bids	assigned_with_jobs
has_markup,	has_overhead_cost	
	has_historial_utilisation	

The manufacturing network is modelled as a subject ResourceHolon:ManufacturingNetwork and the predicates are defined namely is_named, has_network_bids, consists_of_manufacturers and has_contracted_job. The action to be taken on the objects is implemented as predicates bids and accepts_contract.

The customer job is modelled as OrderHolon:Job and the predicates are defined in Table 2. The predicates that take an action on the object are auctions, receives_bid and contracts.

Table 2. Predicates of subject OrderHolon : Job

is_named	has_successful_bid	has_potential_networks
has_network_bids	has_process_plan	has_operation_schedules
has_due_time,	has_customer_goals	has_manufacturer_bids
has_enquiry_time	has_chosen_network	

The offered operation is modelled as a subject `ResourceHolon: Operation` consisting of predicates as shown in Table 3.

Table 3. Predicates of subject ResourceHolon : Operation

is_named	has_actual_used_capacity
has_actual_idle_time	has_conditional_used_capacity
has_operation_cost	has_conditional_idle_time
has_capacity	has_actual_slack_in_schedule
has_quality	has_conditional_slack_in_schedule

And finally, the required operation is modelled as the subject `ProcessPlan: Operation` consisting of predicates namely `is_named` and `has_cycle_time`.

3.4 Interaction Protocol for Network Elements Communication Modelling

Manufacturing network planning is performed through timely communication between manufacturers and the central company of the collaborative manufacturing organization. This communication pattern can be modelled as an interaction protocol. Contract Net Protocol (CNP) is an interaction protocol [18] standardized by FIPA and is based on the auction-based communication model of Dutch auction, English auction and reverse auction. The interaction protocol and ontology complement each other which allow several communication scenarios between manufacturers to be represented. For the purpose of manufacturing network planning and scheduling, an interaction protocol has been developed based on CNP principles as shown in Fig. 2. A case study of a collaborative network organization, the planning and scheduling of jobs in manufacturing networks have also inspired the interaction protocol proposed [7]. The following section describes the different functionalities that the holons must assume during the network formation process.

Manufacturer Bidding for Participation

Assuming that a customer places an order on the central company which is represented by the staff holon in Fig. 2. The function `startOrderAllocation()` involves disaggregating an order into several jobs and each job is assigned to a holon.

The job requests the process plan from the `HMS:ProductHolon` which has a repository of product models and associated process plan. The job holon acquires a process plan upon the response of the function `getProcessPlan(job)`. Function `job.auctioning.callForManufacturerParticipation()` initiates the identification of the manufacturers with the capability of doing the job. This function signals a task announcement [18] to all the manufacturers in the collaborative network organization. Then, through a task bid [18], the manufacturers announce explicitly that they can do at least one operation from the process plan through an activity known as matchmaking as shown in Fig. 3. The notation `shearing>slotting` represents the process plan of `job6`. To achieve the outcome

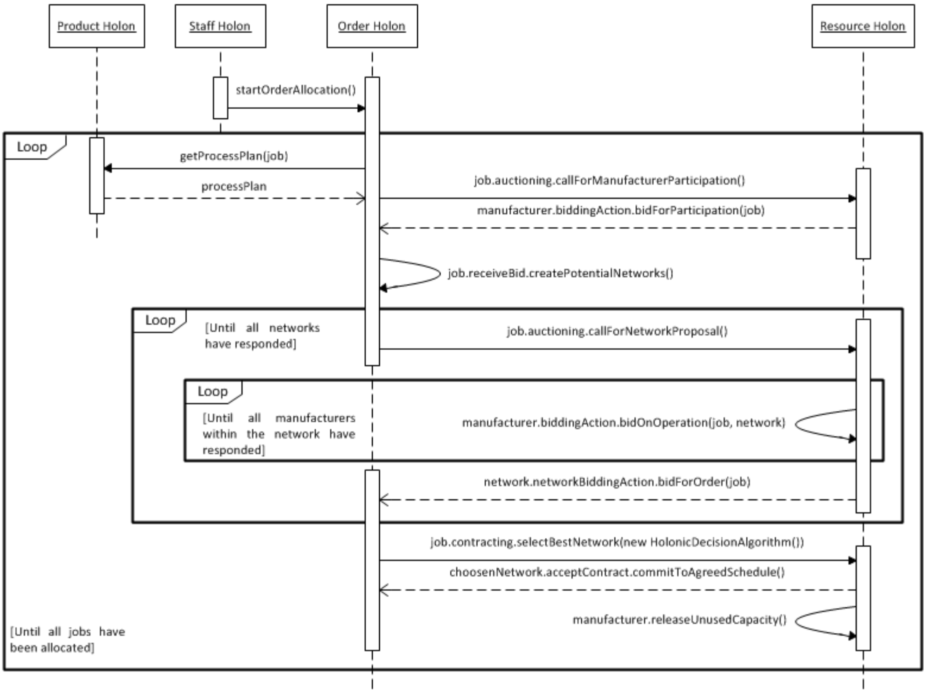


Fig. 2. UML sequence of an adapted Contract Net Protocol for the formation of manufacturing networks and the scheduling of jobs

shown in Fig. 4, the manufacturer holon executes the function `manufacturer.biddingAction.bidForParticipation(job)`.

`Bid45` and `Bid46` are instances of the class `HolonInteraction:ManufacturerBid`. They are partial bids since at this point of the process, the only details known are the manufacturers' offered operations. However, the lead times, costs and quality performance of the manufacturers are not specifically known for `job6`.

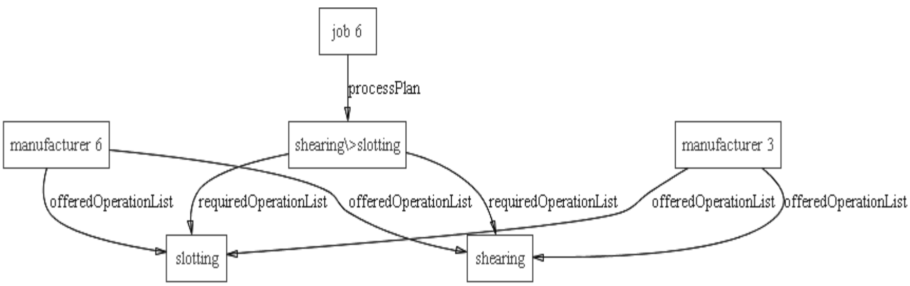


Fig. 3. Ontology of an instance of the matchmaking process between a job and the participating manufacturers

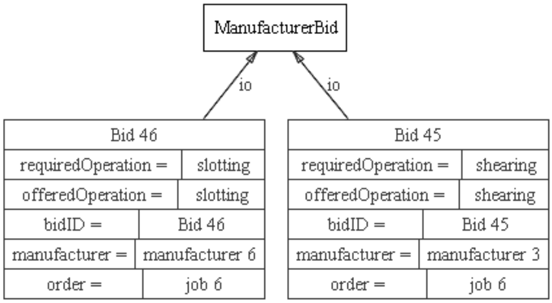


Fig. 4. Partial bid from two manufacturers in response to a call for participation

'Anagrams' of Participating Manufacturers to Create Potential Networks

In order to create all the possible combinations of manufacturing networks that can carry out the process plan of a job, a modified anagram algorithm is used. The new algorithm differs from the original on the basis that it respects the constraints of the process plan i.e. operation dependencies, and it also accounts for the manufacturers that can do two or more operations or even the whole job on their own. In the latter, the term manufacturing network still applies because the central company is always present in any manufacturing network. In Fig. 5, an example is shown of `job6` having four potential manufacturing networks. This outcome is achieved by `job6` implementing function `job.receiveBid.createPotentialNetworks()`.

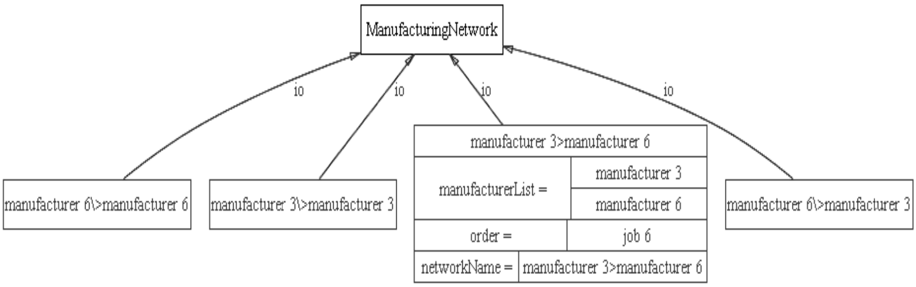


Fig. 5. Ontology of combinatorial manufacturing network formation for a job

Network Bidding for the Job Contract

For the scope of this paper, the scheduling aspect of the job and the costing mechanism of the manufacturers are not described. The focus is on the interaction protocol.

Taking the example of `job6`, there are four potential networks that can be awarded the contract. Function `job.auctioning.callForNetworkProposal()` is used by `job6` to invites the manufacturing networks to complete the initial bid for participation which is shown in Fig. 4. The completed network bid is shown in Fig. 6.

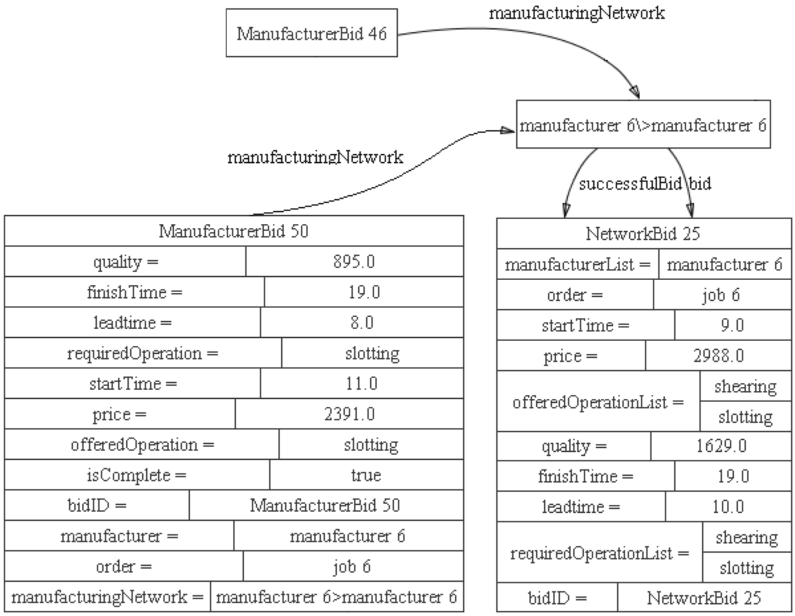


Fig. 6. Network bid results from the collaboration of manufacturers during the bidding process

In network bidding, the structure of the bidding process takes a heterarchical form [16]. In this form, the manufacturers are cooperatively bidding within the network and down the process plan i.e. towards the final operation. The manufacturer bids are securely held within the network. On the other hand, the network bids are not hidden from other networks. Consequently, they will have a tendency to compete for the job while at the same time manufacturers will actually cooperate for the job. During network bidding, the first manufacturer bid is for the first operation in the process plan and from the manufacturer that claimed capability for that operation. The manufacturer bids with `startTime`, `finishTime`, `leadtime`, `quality`, `price` as shown in Fig. 6. This bid is communicated to the next manufacturer that uses the information to create its bid for the second operation in the process plan. This bid-communicate-bid relationship propagates till the last operation in the process plan is bided for. A loop implementation of function `manufacturer.biddingAction.bidOnOperation(job,network)` is carried out.

Distributed Decision Making in Awarding Contract to Manufacturing Network

Based on the `HolonicDecisionAlgorithm()` that filters out bids by lead time, quality and then cost, the best manufacturing network is left [19]. Fig. 7 shows the successful network bid and schedule of job6 at the chosen manufacturing network.

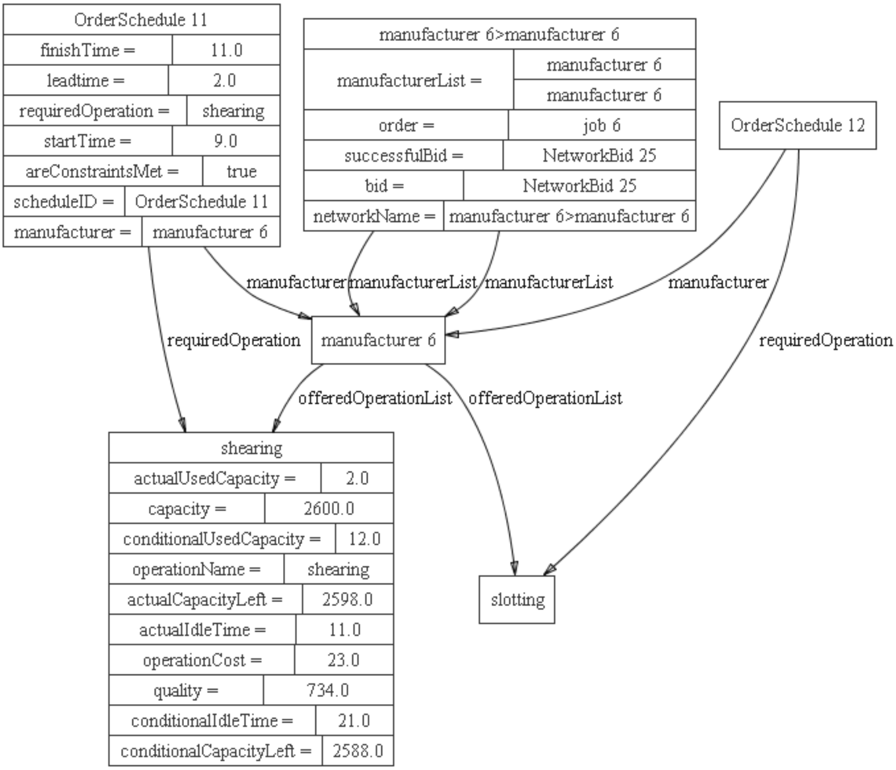


Fig. 7. Ontology of contract schedules between the chosen manufacturing network and the job

The decision algorithm is implemented in each job holon and is triggered by function `job.contractingSelectBestNetwork(new HolonicDecision-Algorithm)`. Therefore, several decision algorithms can be developed and used to reflect the specific requirements of each job. For example `job6`, using the `HolonicDecisionAlgorithm()`, selects the manufacturing network that consists of `manufacturer6`. When the chosen manufacturing network triggers the function `chosenNetwork.acceptContract.commitToAgreedSchedule()`, from each manufacturer's perspective, the `conditionalUsedCapacity` becomes the `actualUsedCapacity` and the `conditionalIdleTime` becomes the `actualIdleTime`. In fact, the other networks which were unsuccessful in the process need to be informed so that each manufacturer can release their conditionally held capacities and this function is implemented by `manufacturer.releaseUnusedCapacity()`. Then, the whole process described in section 3.3 is carried out for the next job until all jobs are allocated.

4 Conclusion

Participating in manufacturing networks becomes a survival strategy for small and medium manufacturers that strive on niche markets. The network involves a managing company that attracts the business and coordinates the delivery of high quality manufacturing service to the customer on behalf of the manufacturers. In this paper the use of the PROSA to create ontology was discussed. A review of the literature shows a gap in the research, on the grounds that the ontology for the auctioning process occurring during the planning and scheduling of networks is not available. Within the scope of manufacturing network planning and scheduling, the ontology is developed to support the objects, their interaction protocol and the knowledge emerging during the auctioning process. The result presented an illustrated 'walk-through' of the activities involved in the interaction protocol using graphs. Also proposed is an interaction protocol applying the principles of contract net protocol for manufacturing network planning and scheduling.

Acknowledgement. The authors would like to thank GFM Srl (Italy) for the sponsorship of the industrial case study of this work.

References

1. Department for business innovation & skills, Statistical Press Release (2010), http://webarchive.nationalarchives.gov.uk/+/http://stats.bis.gov.uk/ed/sme/Stats_Press_release_2009.pdf (accessed online March 13, 2013)
2. Neiuwenhius, P., Wells, P.: The Future of the Automotive Industry: A Guide, BT and Centre for Automotive Industry Research, Economic and Social Research Council, Cardiff (2002)
3. Sourcing Line Computer Economics: Job Outsourcing Statistics (2012), http://www.statisticbrain.com/outsourcing-statistics-by-country/ (accessed online January 29, 2013)
4. Camarinha-Matos, L.M., Afsarmanesh, H., Galeano, N., Molina, A.: Collaborative networked organizations–Concepts and practice in manufacturing enterprises. Computers & Industrial Engineering 57(1), 46–60 (2009)
5. Kock, S., Nisuls, J., Söderqvist, A.: Co-petition: a source of international opportunities in Finnish SMEs. Competitiveness Review: An International Business Journal Incorporating Journal of Global Competitiveness 20(2), 111–125 (2010)
6. Wiendahl, H.P., Lutz, S.: Production in Networks. CIRP Annals–Manufacturing Technology 51(2), 573–586 (2002)
7. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference architecture for holonic manufacturing systems: PROSA. Computers in Industry 37(3), 255–274 (1998)
8. Lin, L.F., et al.: Developing manufacturing ontologism for knowledge reuse in distributed manufacturing environment. International Journal of Production Research 49(2), 343–359 (2011)
9. Jiang, Y., Pang, G., Liu, W.: Research on ontology-based integration of product knowledge for collaborative manufacturing. The International Journal of Advanced Manufacturing Technology 49(9), 1209–1221 (2010)
10. Yan, J., et al.: Ontology of collaborative manufacturing: Alignment of service-oriented framework with service-dominant logic. Expert Systems with Applications 37(3), 2222–2231 (2010)
11. Girt, A., Butte, V.: Engineering Holonic Manufacturing Systems. Computers in Industry 60(6), 428–440 (2009)

12. Lemaignan, S., et al.: MASON: A Proposal for an Ontology of Manufacturing Domain. In: IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and its Applications, pp. 195–200 (2006)
13. PABADISPROMISE consortium: Development of a product and production process description language (2006), http://www.uni-magdeburg.de/iaf/cvs/pabadispromise/dokumente/Del_3_1_Final.pdf (accessed online May 20, 2013)
14. Borgo, S., Leitão, P.: The role of foundational ontologies in manufacturing domain applications. In: Meersman, R., Tari, Z. (eds.) CoopIS/DOA/ODBASE 2004. LNCS, vol. 3290, pp. 670–688. Springer, Heidelberg (2004)
15. Sandholm, T.: An implementation of the contract net protocol based on marginal cost calculations. In: Proceedings of the National Conference on Artificial Intelligence, p. 256. John Wiley & Sons Ltd. (1993)
16. Noy, N.F., McGuiness, D.L.: Ontology development 101: A guide to creating your first ontology, http://protege.stanford.edu/publications/ontology_development/ontology101.html (accessed online May 18, 2013)
17. Giret, A., Botti, V.: Engineering holonic manufacturing systems. Computers in Industry 60(6), 428–440 (2009)
18. Smith, R.G.: The contract net protocol: High-level communication and control in a distributed problem solver. IEEE Transactions on Computers 100(12), 1104–1113 (1980)
19. Jules, G., Saadat, M., Li, N.: Holonic multi-agent system: Simulation of manufacturing network formation. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 686–691. IEEE (2012)

A Method and Application to Simulate and Validate Manufacturing Control Systems Based on a Discrete Manufacturing Simulation Platform

Olivier Roulet-Dubonnet and Per Aage Nyen

Department of Production Technology, SINTEF Raufoss Manufacturing, Trondheim, Norway
{olivier.roulet-dubonnet,per.a.nyen}@sintef.no

Abstract. This paper presents a method and an example application developed to simulate and validate distributed control systems. Using the application, the devices modelled in the manufacturing simulation system, are automatically exposed to the control system with a high level interface similar to the real devices. The devices, including the robots and transport system, can then be transparently operated from the control system as if they were the real devices.

The solution is based on a commercial discrete manufacturing simulation platform called 3DCreate™ from Visual Component, a thin custom interface layer dubbed VCHMS and a multi-agent (holonic) middleware called IceHMS. The VCHMS software communicates in real-time with 3DCreate™ and exposes selected objects and signals to the holonic control system and is also able to translate messages from the holonic system to 3DCreate™ signals.

In addition to its emulation capability, the presented solution has the decisive advantage to reuse the simulation model which is often created earlier for discrete event simulation. The solution has enabled the emulation of a complete production line at GKN Aerospace Norway. The virtual line has been tested and the solution will enable the development and validation of different control strategies from simple pull based systems to complete holonic manufacturing control systems.

Keywords: Holonic Manufacturing System, Multi-Agent System, Simulation, Discrete Manufacturing.

1 Introduction

Traditionally, manufacturing control systems are designed using a top-down approach, implemented in a commercial Manufacturing Execution System (MES) or a development platform, and the logic used is validated using graphical discrete event simulation platform specialised for manufacturing such as Tecnomatix™ from Siemens, Delmia™ from Dassault-Systemes or more recently 3DCreate™ from Visual Component.

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 152–162, 2013.

However, in such simulation platforms, the control logic must be re-implemented, using a dedicated language or graphical interface, thus the validated logic is different and, in most cases, severely simplified compared to the original control system. In addition, when the control system uses more advanced architecture such as the Holonic Manufacturing System paradigm[12], the control logic cannot be implemented inside these platforms which, by their nature, put restrictions on the control logic.

Simulation of the production system using the actual code of the control system and realistically emulated devices would, in this situation, be a decisive improvement. Especially, control systems exhibiting emerging behaviour, such as Holonic Manufacturing Systems, have an even stronger need for verification and should be simulated using the actual agent code, in real-time, refer for example to Vrba and Marík [14, 15].

The key element to run this type of simulation is an emulation platform able to emulate an entire shop floor in real-time, coming with a library of manufacturing devices and processes, and offering an API or protocol to link the emulation system to the control system. A review of the available systems has therefore been conducted ans some key facts are presented in this paper. Refer, for example, to Ruiz et al. [10] for a good classification of manufacturing simulation systems and a large list of available applications.

The world of manufacturing simulation is dominated by a few large commercial products such as Tecnomatix™ and Delmia™ and several smaller ones such as *FlexSim*®. These systems come with a large library of manufacturing devices, which enable the quick development of a model of a shop floor in 3D. They also include graphical interfaces and API to rapidly set up a control flow logic to animate the simulations. In the last years they have started to include the possibility to connect some specific PLC models to the simulation, at least Tecnomatix™ and 3DCreate™ have this possibility. Some newer programs also include an external API which gives some possibilities to control the simulation from external systems, such as 3DCreate™.

There are, at the time of writing, no out-of-the-box open source solutions for emulation of manufacturing shop floors, although efforts are regularly made to develop systems from scratch or extend robotic simulation systems to simulate larger manufacturing systems. For example in Lind and Skavhaug [6] the Blender game engine is used to emulate manufacturing devices. The solution is very flexible, enables the emulation of devices in real-time, but no manufacturing library is available and it requires the modelling of each and every device from scratch. Another custom system is MAST[13], where the authors developed a simulation system targeted at material transportation and build on top of Jade[1]. Simulation systems that are especially designed for a specific multi-agent framework and architecture have also been presented; for example the ABAS WorkBench [5] has been developed specifically for the ABAS architecture. The advantage of such a solution is its deep integration with the framework.

There are several emulation systems targeting the simulation of robots. The Player/Stage[3] is a 2D simulation system targeting AGVs. Gazebo [4] is the 3D successor of Stage and includes physics simulation. Gazebo targets the simulation of robotic systems in general and might be suitable to develop full-scale open-source manufacturing simulation systems. However, as of today, gazebo requires the modelling of every manufacturing device. Openrave[2] is another robotic simulation platform which suffer, in our case, of the same problem of missing manufacturing libraries. These systems have not been tested in detail, thus, they may also suffer from performance problem when emulating larger shop floors.

2 Introducing VCHMS

The main idea in the proposed simulation system is to develop a custom application that takes control of a manufacturing model, implemented in a discrete manufacturing simulation platform, and exposes the devices and their logic to an external (holonic) control system. A prototype system dubbed Visual Component to Holonic Manufacturing System (VCHMS) has been implemented using the simulation platform 3DCreate™ from Visual Component and a flexible multi-agent middleware called IceHMS [9, 8].

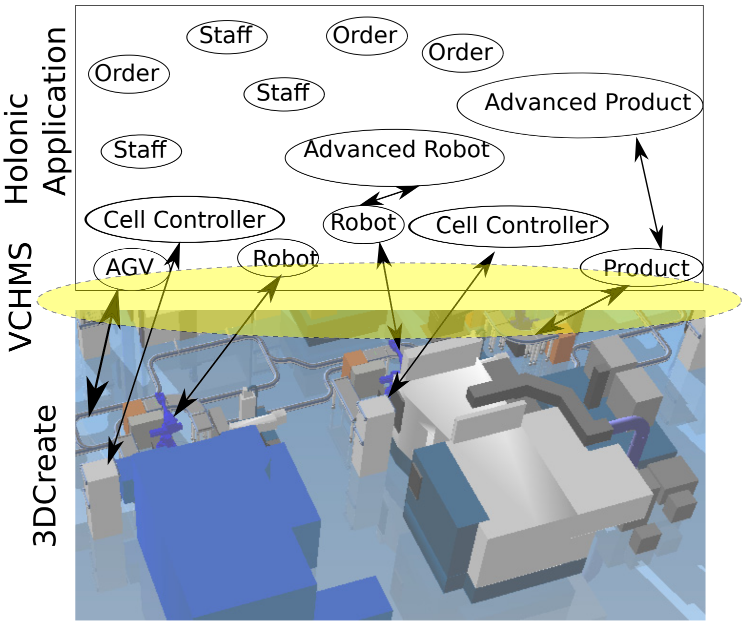


Fig. 1. Sketch of the architecture of a typical simulation implemented using VCHMS

Figure 1 sketches the architecture of a typical holonic system following the PROSA architecture [11]. In this system, the low-level resource- and product-holons (those with an arrow to the 3D model) are automatically created for every device modelled in 3DCreate™. These holons simply expose the devices and the logic already implemented in 3DCreate™. The staff- and order- holons are implemented using a traditional multi-agent platform, such as JADE or IceHMS. The 'advanced product' and 'advanced robot' holons, have additional logic, implemented in the holonic application, to take directly part in the usual life of agents such as collaboration, bidding for jobs and negotiation.

The realisation of the simulation is depending on the manufacturing simulation system to offer an API to control the models in the simulation and receive events from them. The API must also be able to handle the large number of devices and events which are characteristic of manufacturing shop floors. Especially the robots and transport system API need a high level of flexibility since they are the glue between the elements of the simulation.

3DCreate™ is a commercial software targeting both discrete manufacturing simulation and off-line robot programming. Visual Components' offering is relatively recent but seems to be successful and has tight connections with industrial robot providers such as Stäubli and Kuka. Before starting the development of VCHMS, several discrete event simulation platforms were evaluated. 3DCreate™ was selected due to the availability of a well thought external API (based on DCOM) especially the API related to robot control, see Sec. 3.3, and its discrete event simulation capabilities inclusive its modern graphical interface. These features may exist in other commercial packages that may also be suitable to the task.

The solution is targeted at simulating large production sites: the interfaces offered by emulated holons are, per default, of relatively high level, such as start, stop, number of pallets, etc. To the authors' experience, this is perfectly in accordance with the control requirements of industrial MES Systems. When more fine-grained control of the devices is required, the interface can be customized either in VCHMS itself or by creating a higher level holon in the multi-agent system. If the case requires lower level simulation like real-time robot control or metal-cutting simulations, other specialized systems should be sought.

3 Implementation

Figure 2 sketches the main objects in a simulation system implemented with VCHMS. VCHMS is a background service and does not include any graphical interface, although a GUI could be developed or it can be integrated as a 3DCreate™ plugin or command. At startup VCHMS queries Visual Component for its running objects and publishes selected objects as holons, through the IceHMS framework. Most objects are published as-is with an interface created at run-time from their available 3DCreate™ signals, but some objects, such as robots and the simulation object, have additional object-specific logic and methods.

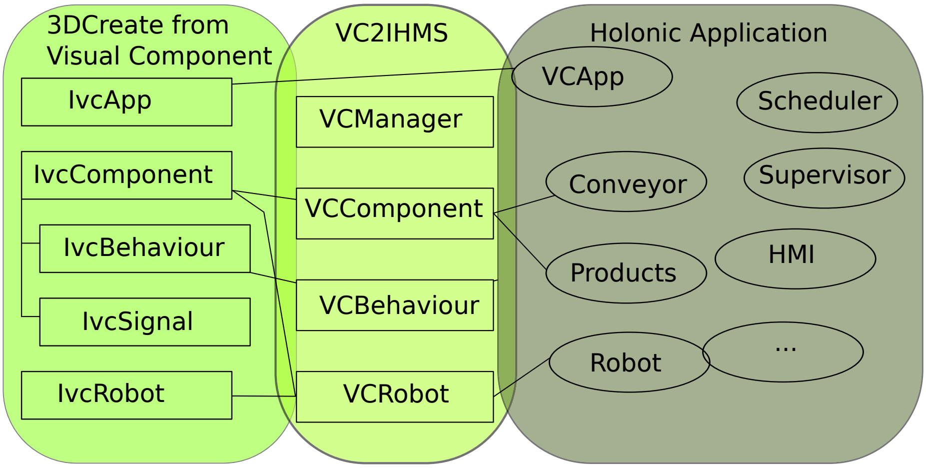


Fig. 2. Overview of the main objects in a simulation using VCHMS

Since the external API of 3DCreate™ is based on DCOM, VCHMS has been developed in C# which is a relatively efficient, modern language, with good DCOM support. C# is also fully supported by IceHMS. VCHMS is small and is currently around 1000 lines of code. Apart from the Visual Component DCOM, VCHMS does not have any other platform specific requirements. VCHMS is published as open source and available for everyone to use and modify [7].

Although the presented concept is independent of any specific multi-agent system, the current implementation is tightly coupled to IceHMS. IceHMS is a multi-agent middleware, comparable to Jade but supporting several programming languages and targeted at the development of situated Multi-Agent Systems, see Roulet-Dubonnet et al. [9].

3.1 Components

Components are the base element exposed to the holonic system. In 3DCreate™, every manufacturing device is a component, for example conveyors, robots and CNC machines. Per default, all 3DCreate™ components are exposed, either as component holons or as specialized types, for example robots, with a specific interface, see fig. 2.

3DCreate™ components also contain properties, and objects called behaviours. Properties are exposed as Holon properties and can be read and set from the holonic system. 3DCreate™ behaviours are various objects ranging from Python scripts to kinematics, through other 3DCreate™ specific objects. Behaviours, as their name suggest, define the behaviour of the component. Behaviours and their properties can also be read and set from the holonic system. See the interface definition in Listing 1.1.

```
interface PropertyList {
    string getProperty(string name);
    StringSeq getPropertyList();
    void setProperty(string name, string val);
} ;

interface Behaviour extends PropertyList{
};

interface Component extends Holon, PropertyList {
    StringSeq getBehaviourList();
    Behaviour* getBehaviour(string name);
};
```

Listing 1.1. The 3DCreate™ specific interface definition of component(device) Holons in VCHMS

3.2 Signals

In 3DCreate™ objects communicate using signals or IOs. Signals are *behaviours* of components. Signals are not exposed directly to the holonic system, but for every signal a topic is published which can be subscribed by holons. Thus, when a signal inside a component is activated, all subscribed holons get the signal information. 3DCreate™ defines several types of signals ranging from boolean to signals having links to internal 3DCreate™ objects.

Component holons can, through messages from other holons, activate their own signals. For example a boolean message of the type *Signal* with the name MyBooleanSignal and a value will activate the component signal of the same name. Signals are the default method to communicate with devices inside 3DCreate™, although properties (Sec. 3.1) can be used.

3.3 Robots

Robot holons are specialized components. Robot holons can be controlled as generic components with robot programs implemented inside 3DCreate™ and the execution triggered by signals. This is how industrial robots usually communicate: with programs inside the controllers and limited external communication. However, it is often much easier and more flexible to program the robots from the holonic world using a generic programming language. Robots in 3DCreate™ are therefore exposed as robot holons with a set of simple methods such as 'movel' which move the robot tool coordinate system to a given pose in the current coordinate system, 'movej' which moves the robot joints, and 'grasp' and 'release' which grasp or release the closest object(s) in the simulation model, see Lst. 1.2.

```
interface GenericRobot extends Component {
    void set_csys(CSYS cref);
    ["ami"] void movel(DoubleSeq pose, double acc, double vel);
    DoubleSeq getl(); # return current position in set csys
    ["ami"] void movej(DoubleSeq joints, double acc, double vel);
    DoubleSeq getj(); # return joint values in radian
    bool is_program_running();
    void set_digital_out(int nb, bool val);
    ...
    void set_tool(int tool);
    void set_tcp(DoubleSeq tcp);
    void grasp(); // commodity method
    void release(); // commodity method
    }
```

Listing 1.2. The default interface definition of a robot in VCHMS

Internally, the robot implementation uses the IvcRobot interface of the Visual Component platform, which allows the integration of a custom robot controller into 3DCreate™. The current implementation has a rudimentary controller which is queried by the simulation when necessary, but more advanced implementations with integrated path planning are possible.

3.4 Others

In addition to manufacturing objects, VCHMS also creates holons which are simulation specific and are used only to control the simulation. In particular, VCHMS creates a unique VCApplication Holon which can start, stop and reset the simulation, and component-creator objects which can control the creation of products on the production line.

4 Experiments and Results

At the time of writing, VCHMS has been used in two applications. One is a simple pick and place application which has been used to develop the software and the other one is a complete production line at GKN Aerospace Norway, with a simple pull-based agent control system.

4.1 Example Pick and Place Application

To develop VCHMS a simple pick and place application has been created. In this application, the model is composed of 2 conveyors, 2 robots and a special component generating products on the first conveyor. The first conveyor has a sensor which sends a signal. Usually, when simulating in 3DCreate™, one would create interfaces between the robot and the first conveyor, connect them

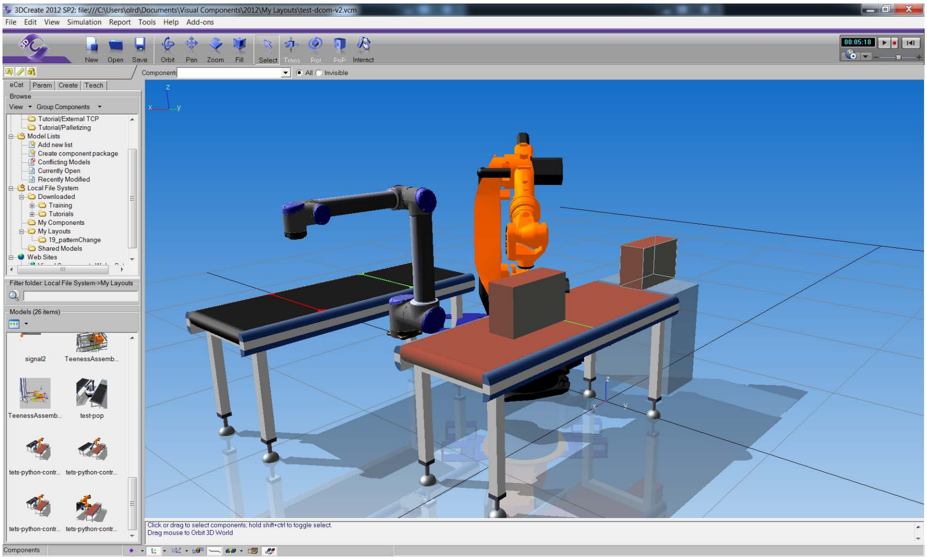


Fig. 3. Pick and place application used to develop VCHMS

and create a robot program in 3DCreate™ using the internal language or the internal Python scripting to catch signals and move the robot.

In this application, the component-creator object creates products which start moving on the first conveyor. When the products pass the sensor, they generate a signal with a link to themselves. If a product is not picked, it disappears at the end of the first conveyor.

At startup VCHMS exposes the robot and the conveyors to the holonic control system. At runtime, when products are created, VCHMS also exposes them also as holons.

The workflow is then entirely implemented in the holonic control system, outside 3DCreate™, and is the following:

1. In IceHMS a cell controller holon is created. This holon subscribes to messages from the conveyor and get a proxy to the robot holons.
2. When the signal on the first conveyor is triggered, the conveyor publishes a message with the product's component name.
3. The cell controller holon receives the signal, and controls one of the robots to pick the product on the first conveyor and place it on the second conveyor using the 'movel' and 'movej' methods of the robot holon, see Sec. 3.3.

4.2 Emulation of a Production Line at GKN Aerospace

The IHAP (Innovative Highly Automated Production of turbine vanes) project is a cooperative research project run by GKN Aerospace Norway which aim is to develop a new production system for turbine vanes enabling one-piece production and one-piece flow with reduced manpower.

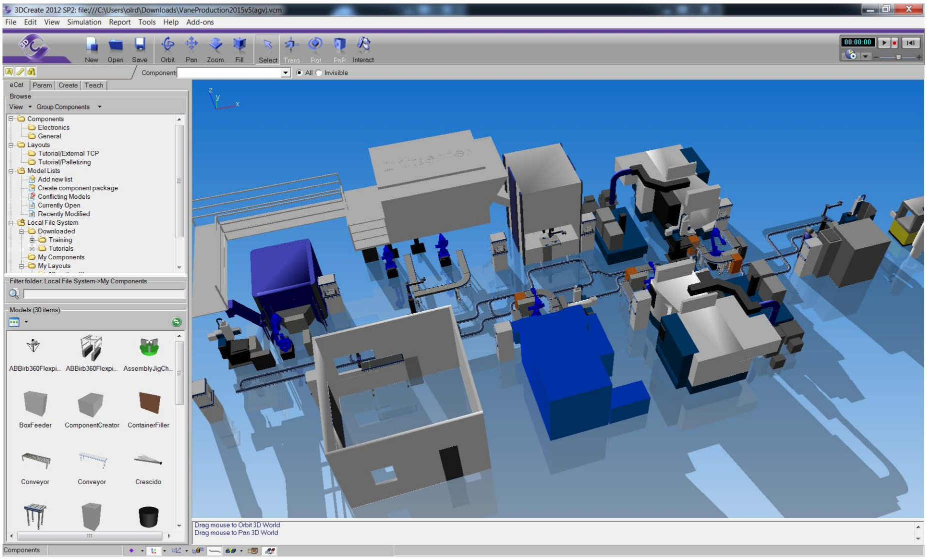


Fig. 4. A part of the vanes production line at GKN Aerospace Norway simulated in 3DCreate^{TM}

For the IHAP project, several models, representing future alternatives and past iterations of the production line, have been created for evaluating concepts and layouts, and carry out production flow analyses. These models were developed in the discrete manufacturing simulation 3DCreate^{TM}, see Fig. 4 for a screenshot of one model. For this work, the control logic was implemented entirely in 3DCreate^{TM} using the dedicated graphical interface and the integrated Python scripting.

However, to support the development of the next generation control system at GKN Aerospace, a more flexible solution was necessary. The solution needed to support the emulation of the entire production line, and its control by the production control code, in order to demonstrate, test and validate all the components of the distributed control system and its logic. This led to the completion of the development of VCHMS, the idea of which had been laying around a couple of years.

Using VCHMS, all the devices and all the signals in the virtual production line, are automatically exposed as holons and messages. The use of VCHMS on one model of the production line led to the effortless emulation of 7 CNC machining cells, 14 robots, an automatic storage and retrieval system (ASRS), a transport system and the emission and receiving of hundreds of signals. Since the logic of the devices had already been implemented in 3DCreate^{TM}, the entire simulation can be controlled by just a few messages. Or, when desired, lower level control is available by controlling the robots or other devices directly, and subscribing to lower level signals.

At the time of writting, a simple pull based multi-agent system has been implemented as a test application. In this application one agent is created per cell controller. Each agent monitors its queues, when the number of products in the entry queue is lower than a chosen level, a pull request is sent. This system is very simple and is only to be seen as an application to test the control of the entire simulated production line and is to be replaced by different control applications, from simple pull based systems to complete holonic manufacturing control systems.

5 Conclusion

The system presented in this paper enables the simulation and therefore the validation of control systems for manufacturing. This is especially valuable when the control system allows for emerging behaviour, whose logic can usually not be implemented in discrete manufacturing simulation platforms.

In addition the solution allows for the re-use of 3D models, including their logic, developed for discrete manufacturing simulation. Since discrete manufacturing simulation is usually done earlier to the production control system design, this results in a much shorter design and validation process.

The system has been tested both for simple pick and place scenarios and for the emulation of several versions of a complete production line at GKN Aerospace. The solution will enable the testing and validation of different control strategies of the production line, from simple pull based systems to complete holonic manufacturing control systems.

Acknowledgments. The work presented in this paper has been realized for the IHAP project with support from the CRI Norman. IHAP(Innovative Highly Automated Production of turbine vanes) is an industrial research project run by GKN Aerospace Norway in cooperation with SINTEF, the Norwegian University og Science and Technology, Trondrud Engineering, Jærtek and Kongsberg Terotech and supported by the research council of Norway.

References

1. Bellifemine, F., Poggi, A., Rimassa, R.G.: JADE - a FIPA-compliant agent framework. In: Proceedings of the Practical Applications of Intelligent Agents. The Practical Application Company Ltd. (1999) 1
2. Diankov, R.: Automated Construction of Robotic Manipulation Programs. Ph.D. thesis, Carnegie Mellon University, Robotics Institute (August 2010), `http://www.programmingvision.com/rosen_diankov_thesis.pdf` 1
3. Gerkey, B., Vaughan, R., Howard, A.: The player/stage project: Tools for multi-robot and distributed sensor systems. In: Proceedings of the 11th International Conference on Advanced Robotics, pp. 317–323. Citeseer (2003) 1
4. Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004), vol. 3, pp. 2149–2154. IEEE (2004) 1

5. Lastra, J.L.M., Torres, E.L., Colombo, A.W.: A 3D visualization and simulation framework for intelligent physical agents. In: Mařík, V., Brennan, R.W., Pěchouček, M. (eds.) HoloMAS 2005. LNCS (LNAI), vol. 3593, pp. 23–38. Springer, Heidelberg (2005) 1
6. Lind, M., Skavhaug, A.: Using the blender game engine for real-time emulation of production devices. International Journal of Production Research, 1–17 (2011) 1
7. Roulet-Dubonnet, O.: VC2HMS: Visual Component to IceHMS Interface (2013), `https://github.com/oroulet/vc2hms/` 3
8. Roulet-Dubonnet, O., Lind, M.: IceHMS: A Holonic Middleware (2010), `https://github.com/oroulet/icehms/` 2
9. Roulet-Dubonnet, O., Lind, M., Skavhaug, A.: IceHMS, A middleware for distributed control of manufacturing systems. In: Mařík, V., Martinez Lastra, J.L., Skobelev, P. (eds.) HoloMAS 2013. LNCS (LNAI), vol. 8062, pp. 95–105. Springer, Heidelberg (2013) 2, 3
10. Ruiz, N., Giret, A., Botti, V.: Towards an agent-based simulation tool for manufacturing systems. In: IEEE Conference on Emerging Technologies and Factory Automation, ETFA 2006, pp. 797–804 (September 2006) 1
11. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference architecture for holonic manufacturing systems: PROSA. Computers in Industry 37(3), 255–274 (1998) 2
12. Van Brussel, H.: Holonic Manufacturing Systems The Vision Matching the Problem. In: Proceedings of the 1st European Conference on Holonic Manufacturing Systems, Hannover, Germany, IFW-Hannover (1994) 1
13. Vrba, P.: Mast: manufacturing agent simulation tool. In: Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation, ETFA 2003, vol. 1, pp. 282–287 (2003) 1
14. Vrba, P., Marík, V.: From holonic control to virtual enterprises: The multi-agent approach. In: The Industrial Information Technology–Handbook, pp. 107–109 (2005) 1
15. Vrba, P., Marík, V.: Simulation in agent-based manufacturing control systems. In: 2005 IEEE International Conference on Systems, Man and Cybernetics, vol. 2, pp. 1718–1723 (October 2005) 1

Traffic Radar: A Holonic Traffic Coordination System Using PROSA++ and D-MAS

Johan Philips, Bart Saint Germain, Jan Van Belle, and Paul Valckenaers

Department of Mechanical Engineering, KU Leuven,
Celestijnenlaan 300B, 3001 Leuven, Belgium
{first.last}@mech.kuleuven.be
http://www.mech.kuleuven.be

Abstract. This paper presents **Traffic Radar**, a **holonic** traffic coordination system focusing on the distributed and dynamic nature of traffic systems. Two main architectural assets enable distributed real-time coordination, the holonic **PROSA++ architecture** and the **delegate multi-agent system** (D-MAS) pattern. Well proven and accurate first-order traffic models are used to model local traffic behaviour. Link and node holons encapsulate local traffic models and offer services to other holons and D-MAS ants in the environment. Early experiments with the Traffic Radar platform show its ability to forecast traffic flows and densities based on individual user intentions. Moreover, they show the ability to explore different routing solutions incorporating traffic density forecasts.

Keywords: Holonic system, traffic models.

1 Introduction

A traffic system comprises a set of autonomous entities. All entities have, to some extent, a degree of freedom: (1) vehicle drivers have the freedom to select their personal route and timing; (2) traffic infrastructure cannot be centralised entirely. Still it is opportune to coordinate these *selfish* users. This is mainly achieved by providing information and incentives, such as personal guidance or cost estimation.

The *Traffic Radar* is a holonic traffic coordination system which creates and maintains a short-term forecast of traffic flows and densities. The short-term forecast is characterised by its high fidelity through a high participation of individual traffic users who propagate their intentions. It provides a unique possibility to explore different routings and timings. The explored alternatives are the key information to guide and steer users towards a system optimum.

Traffic is also characterised by its fluctuating state. Disturbances like car accidents, weather conditions or changing traffic demand influence heavily the traffic flow and density.

The Traffic Radar's holonic architecture is a natural mapping of the traffic infrastructure and its users. The choice of this holonic architecture is further elaborated in the following section.

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 163–174, 2013.

1.1 Holonic Architecture in Traffic Context

Two main architectural assets enable distributed real-time coordination, the holonic PROSA++ architecture and the delegate multi-agent system (D-MAS) pattern. In the context of traffic, a holon is defined as an autonomous and co-operative building block of a traffic control system for informing and guiding traffic entities.

Using a holonic architecture requires an identification of different holon types. Responsibilities need to be assigned to each holon type and relationships between holons need to be defined clearly. PROSA++ is a reference architecture for holonic architectures that reduces impact of changes in decision making by separating concerns. This reference architecture allows to [11,1]:

- Separate *resource* (traffic infrastructure) aspects from *product* (trip planning and driving to its destination) specific aspects. Typical for traffic control is the difference in goal between *selfish* traffic users and traffic infrastructure as system-wide optimiser.
- Separate necessary modules, which are generic, from optional modules, which can be domain specific. Different holons in PROSA++ all hide specific technical details from each other.
- Separate structural aspects of the architecture from algorithmic aspects. Existing scheduling and planning algorithms can be integrated without affecting the basic architecture.

Traffic is characterised by its distributed and dynamic nature. A holon needs to be capable of making decisions that are adapted to events happening in the environment, i.e. the traffic network. The delegate MAS pattern allows holons to exploit detailed and up-to-date information in this environment and to adapt their intentions to these events.

2 PROSA++ for Traffic Coordination

The holonic architecture, PROSA++, is adopted from a well-studied and widely used architecture in manufacturing control systems, PROSA [1]. The PROSA architecture was originally developed for the manufacturing domain. It has also been applied in other application domains (railway systems [3], logistic systems [10], robotic systems [7], and others [9]). This architecture is further elaborated towards PROSA++ which appeared to be well suited for traffic control systems.

2.1 PROSA++ Reference Architecture

The PROSA++ architecture identifies four holon types as indicated in Figure 1. A distinction is made between products and resources. Products relate to any activity in the world of interest, in this case traffic users. Resources relate to enabling entities in the world of interest, in this case traffic infrastructure entities, e.g., links and nodes.

Note that due to the origin of PROSA++ (manufacturing), the terminology product and resource may seem confusing. Nevertheless, their concepts remain valid and have been proven outside the manufacturing domain [9].

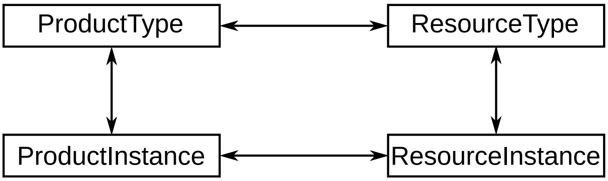


Fig. 1. Holonic reference architecture PROSA++, based on well-studied and widely used architecture in manufacturing control systems, PROSA [1]

The `ResourceInstance` holon reflects a specific part of the traffic infrastructure. This holon contains information about the physical entity it reflects, present and future states and what-if functionality. We consider two classes: route infrastructure, i.e. nodes and links, and multi-modal transport, such as trams and buses.

The class information is encapsulated into the `ResourceInstance` holon. The same interface is used to the remaining system. Preferences to use one or another `ResourceInstance` holon are expressed in terms of capability (e.g. amount of luggage), availability and trust (reputation from one user towards a `ResourceInstance` holon).

A `ResourceType` holon holds policies regarding a specific resource entity or a group of resources. Examples of these policies are: maximum and minimum speed on a link or the presence of bus lanes. Policies are communicated to `ResourceInstance` holons in order to ensure a correct and desired behaviour on the traffic network entity.

The `ProductInstance` holon corresponds to a request for a trip in the traffic network, originating from a traffic user. This holon is responsible for planning and guiding the assigned trip correctly and on time. The `ProductInstance` holon searches and evaluates candidate routes and potential multi-modal alternatives. A selection of candidates is presented to the user while the final selection - the intention - is the responsibility of the user. The `ProductInstance` holon reflects this decision to other holons.

`ProductType` holons hold all policies regarding a traffic user or a group of users. These policies can be various, some examples:

– Preferences regarding multi-modal transport, vehicle availability.
– Preferences regarding distance.
– Preferences regarding route types (e.g. scenic route), stops allowed or preferred.

These policies are exchanged with the `ProductInstance` holon to guide the search for good journeys.

3 D-MAS for Traffic Coordination

Delegate multi-agent system (D-MAS) is an architectural pattern that allows a holon to delegate a responsibility to a swarm of lightweight *ant agents* to support this holon in fulfilling its functions [6,11]. The issuing holon can delegate multiple responsibilities, each of them applying the delegate MAS pattern. The holon may use a combination of delegate multi-agent systems to handle a single responsibility. The D-MAS may also provide services to other holons.

The *D-MAS* pattern translates insights from the food foraging behaviour in ant colonies into the software design [8].

- **Refresh-and-evaporate**: ants deposit pheromone trails that evaporate unless refreshed by ants walking along such a trail. This translates into: All information in the traffic management system that is subject to real-world dynamics has a finite lifespan. For instance, trip reservations need to be reconfirmed regularly or they are discarded by the resource (e.g. parking space) concerned.
- The **environment** contributes to the solution: ants deposit their pheromone trails on the real-world environment, allowing them to cope with almost any geometrical complexity by means of a single simple procedure. Translating this mechanism, an environment is created that mirrors the world-of-interest in software (physical traffic infrastructure), and environment components will be made intelligent/cognitive as needed or opportune.
- **Swarming**: from simple ant behaviours emerges sophisticated colony behaviour. However, the colony needs numerous cheap ants to achieve this. In the model, traffic users create swarms of lightweight agents - called ant agents - that travel virtually across the environment. Because they are virtual entities, ant agents are cheap and can be numerous. These swarms, performing services on behalf of holons, are called a D-MAS.
- **Computational efficiency**: our ant-like design has a low-polynomial computational complexity in function of the effort needed for the primitive actions of virtual travel through the environment.

4 Traffic Models

This section explains briefly the traffic models used by link and node holons to predict traffic flow. Local traffic models comply with simplified kinematic wave theory used in link transmission models [2,4,13]. The fixed point formulation of the link transmission model [2,4,13] enables an iterative approach and justifies the usage in a distributed and real-time context.

In the link transmission model individual vehicles are aggregated into a continuous vehicle flow (represented by cumulative vehicle numbers). Figure 2 depicts a schematic representation of a link by the cumulative flow at the beginning of the link (upstream) and at the end of the link (downstream). The cumulative flow at the end of the link is shifted in time as it takes time to travel though

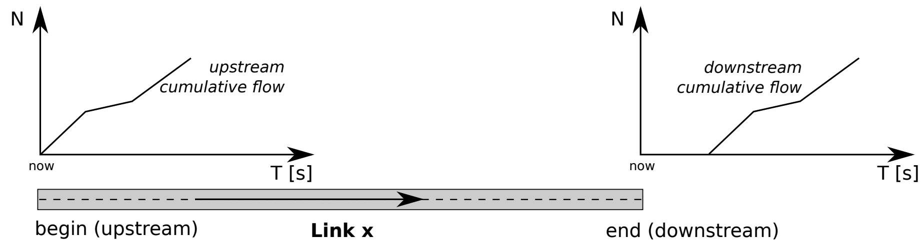


Fig. 2. Representation of a link by begin and end cumulative flow

the link. These cumulative flows are calculated iteratively and are adjusted whenever the context changes or new constraints are appearing from changed traffic demand or traffic supply, e.g. traffic jam and queue spill back due to insufficient downstream capacity or an accident.

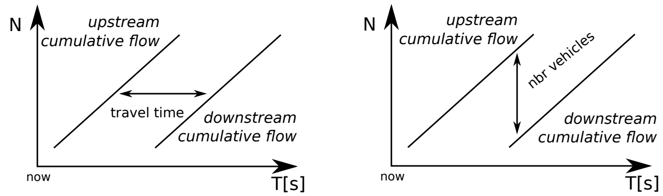


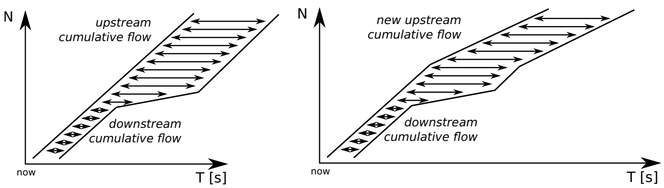
Fig. 3. Properties of link representation by cumulative flows

The representation by cumulative vehicle numbers holds some interesting basic properties: link travel time and total number of vehicles on the link (figure 3). The horizontal distance between a same cumulative count at the beginning of the link and at the end of the link indicates the travel time for the time at which the entrance cumulative count is reached. The vertical distance between begin and end cumulative flow indicates total number of vehicles at that time.

4.1 Forward Propagation

This model propagates traffic flow from upstream to downstream within one link. Moreover, the model calculates downstream cumulative flow by adding the appropriate travel time (valid at that point in time) to each point in the upstream cumulative flow. In case there are no previous iterations, free flow travel time is used, i.e. the time to cross the link at maximum speed. If there was a previous iteration, travel time is calculated by comparing up- and downstream cumulative flow from the previous iteration as explained in section 4.

Figure 4 shows forward propagation of a changed upstream cumulative flow in iteration i+1. Travel times are adopted from iteration i and result in an updated downstream cumulative flow.

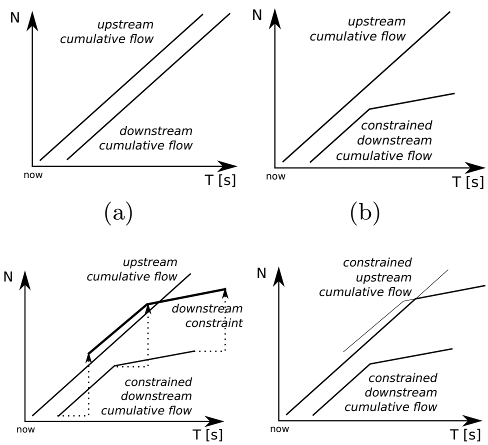


(a) iteration i (b) iteration i+1

Fig. 4. Forward propagation

4.2 Backward Propagation

Here, the model propagates traffic flow from downstream to upstream within one link. Two steps can be identified in this model: upstream constraint calculation and applying this constraint to the original upstream cumulative flow.



(a) (b)

(c) (d)

Fig. 5. Backward propagation

This section explains the model based on an scenario (Figure 5). Please refer to [13] for a full understanding of this model.

The scenario consists of four steps:

1. The link is in free flow state: downstream cumulative flow is shifted with free flow time as compared to the upstream cumulative flow (figure 5 a);
2. A car accident on a downstream link gives rise to a constrained downstream cumulative flow (figure 5 b);
3. The updated downstream cumulative flow results in an upstream constraint (figure 5 c). This constraint represents the maximum flow which can enter at the beginning of the link. The maximum entrance flow is calculated by

shifting the upstream cumulative with the spill back time (i.e. the minimum time a queue at the end of the link needs to reach the beginning of the link) and augmented by the numbers of cars at the link in case of traffic jam;
4. The original upstream cumulative flow is constrained by the maximum flow defined by the propagated downstream flow (figure 5 d). The new upstream cumulative flow is formed by the minimum of the original cumulative flow and the maximum flow evaluated in all points.

4.3 Capacity Constraint

A node or link has a limited capacity. This corresponds to a maximum gradient in cumulative flows. This maximum gradient is applied to the cumulative flow by lowering the flow to the maximum capacity if the original flow is too high. Figure 6 shows the original cumulative flow and the constrained cumulative. Flow from time t1 until t2 is below the maximum capacity and does not need to be constrained. After time t2 flow gets higher and is constrained accordingly until the intersection from the gradient with the original cumulative flow: t4. This procedure is repeated until the full cumulative flow is constrained.

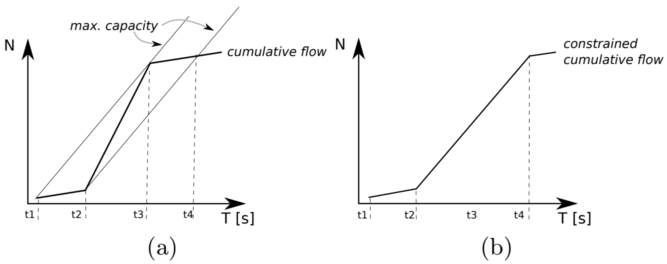


Fig. 6. Flow constrained by capacity

5 Implementation: Traffic Radar

The holonic concepts provided by PROSA++ and D-MAS have been applied to traffic coordination. This has led to the implementation of the *Traffic Radar*, which uses PROSA++ to model the traffic infrastructure and several D-MASs to forecast traffic flows.

5.1 Erlang

Erlang provides massively parallel and distributed processing. Benchmark tests with up to one million processes ensure the creation of large traffic networks with many ants dropping pheromones on links and nodes. Moreover, every erlang process is an *actor*. Many ideas in the actor model [5] are now applied in multi-agent systems [12]. Erlang, therefore, is excellently suited for holonic multi-agent systems and complies with the dynamic and distributed nature of traffic.

5.2 Overview

In the holonic architecture, the traffic infrastructure, i.e. the network, is represented by a set of two distinct `ResourceType` holons: `Link` holons and `Node` holons. Furthermore `Vehicle` holons, represented by `ProductInstance` holons, correspond to trip requests in the traffic network, originating from traffic users. Table 1 provides an overview of the different holons and their responsibilities.

Table 1. Overview of the different holons in the Traffic Radar application, their life cycles and services and which D-MAS they use

Holon	**Life cycle**	**Services**	**D-MAS**
Link	update state forward flow propagation backward queue propagation	execute scenario proclaim scenario propagate scenario	none
Node	detect capacity constraints	execute scenario proclaim scenario propagate scenario	flow D-MAS
Vehicle	update state select intention (auto)	find solutions select intention (manual)	exploring D-MAS intention D-MAS

5.3 Link Holon

The life cycle of a `Link` holon is characterised by three distinct actions. Firstly, dynamic map information, received by e.g. a traffic monitoring system, is used to update its state. Secondly, based on this dynamic map information, such as traffic density, propagation of flows and cumulative functions are calculated. Thirdly, back propagation of queues, e.g. due to bottlenecks, is performed.

`Link` holons also provide services to other holons. The `ExecuteScenario` service can be used to perform a what-if scenario on the holon. The service returns travel time on the respective link given an arrival time and estimated traffic density. The `ProclaimScenario` service enables holons to calculate the travel time and save their intention indicating at what time they would arrive and depart on the link. Finally, the `PropagateScenario` service offers the ability to adapt upstream or downstream flow on the link. Consequently, adapted flow is propagated within the link. Adapting upstream flow results in forward propagation, while downstream flow adaptations result in backward propagation.

5.4 Node Holon

The life cycle of a `Node` holon consists of detecting capacity constraints. If flow inconsistency between upstream and downstream links in a particular node occurs, e.g. due to an accident or structural bottleneck, flow has to be adapted. To propagate traffic flow constraints both upstream and downstream, the `Node` holon creates flow ants. This propagation models spill back over links and nodes.

The services offered by `Node` holons are identical to those of `Link` holons.

5.5 Vehicle Holon

`Vehicle` holons represent users driving through the traffic network and, therefore, also virtually move through the `Link` and `Node` holons. To indicate where a `Vehicle` holon resides and where it is moving to, it sends out two different D-MASs: exploring D-MAS and intention D-MAS. These D-MASs drop pheromones on the traffic infrastructure holons to respectively search and proclaim the route of the relevant `Vehicle` holon from its origin to destination.

The life cycle of this holon also contains, besides exploring and intention ant creation, functionality to update its state, containing information such as its origin and destination, the current link or node it resides on, and vehicle properties. If configured so, it also automatically selects a route out of solutions, provided by the exploring ants.

The `Vehicle` holon offers two services to other holons or external users. Firstly, the set of routing solutions, from its origin to destination, can be requested. Secondly, a user can manually select one of these solutions to commit to. In this case the automatic intention selection is disabled.

5.6 D-MAS Implementation

All ants in the different D-MAS have the same life cycle: (1) a scenario is created based on their current state, (2) this scenario is executed on the resource, on which the ant currently resides, and (3) this process is repeated until an end condition has reached. Both scenario creation and execution are D-MAS specific.

In Erlang, these steps are represented by functions, which can be passed to the ant state at creation, reducing the life cycle code to a few lines, as depicted in Listing 1.1.

```
loop(Ant = #ant{createScenario = CS, executeScenario = ES,
    history = H, state = S}) ->
     Scenario = CS(S,H),
     NewState = ES(Scenario),
     NewState =:= ?undefined orelse
     loop(Ant#ant{history = [S|H],state = NewState}).
```

Listing 1.1. Erlang code for generic ant life cycle

5.7 Exploring D-MAS

The exploring D-MAS, used by `Vehicle` holons, move through traffic infrastructure holons to search for a routing solution from the vehicle's origin to destination. On each holon they visit, the `ExecuteScenario` service is executed to determine travel time on that holon, based on provided arrival time, which is typically in the near future, and the traffic at that time. This service uses the travel time model discussed in section 4.

5.8 Intention D-MAS

`Vehicle` holons also deploy intention ants through the traffic network. These ants receive a selected solution, chosen by their creator, from the solutions gathered by the exploring ants. The intention ants drop pheromones on the relevant holons to proclaim the `Vehicle` holon's presence on these holons at a certain time. To accomplish this, intention ants employ the `ProclaimScenario` service of the currently occupied holon. This service determines travel time, similarly to the `ExecuteScenario` service, based on the travel time model.

The difference with exploring ants is twofold: (1) a solution, i.e. a route, is provided at creation and (2) pheromones are dropped on traffic infrastructure holons to indicate when the vehicle would be where.

5.9 Flow D-MAS

The last D-MAS ants, used in the Traffic Radar application, are created by `Node` holons. As discussed, flow ant creation is triggered by flow inconsistencies in a node. Flow ants make use of the `PropagateScenario` to adapt upstream or downstream flow. By executing this scenario, a new constraint can exist, in which case the flow ant continues to move upstream or downstream. If no new constraint is formed, the flow ant stops propagating.

6 Experiments

This section presents early experiments with the Traffic Radar platform. The experiments show the ability to forecast traffic flows and densities based on individual user intentions.

If the accumulated traffic on a link remains below the link's maximum capacity, the flow is not constrained and a free flow model is used to calculate the cumulative at the end of the link. Figure 7 depicts both cumulative flow at the beginning and end of the link. In this case, no constraints are applied, resulting in free flow. The cumulative function is just shifted into the time.

In contrast, Figure 8 shows an example, in which too many vehicles enter a link and constraints are propagated. This process consists of three propagation waves:

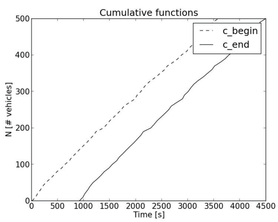
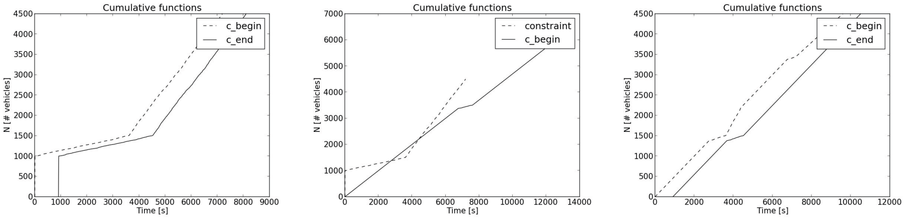


Fig. 7. Flow not constrained by downstream link



(a) Forward propagation (b) Backward propagation (c) Forward propagation

Fig. 8. Cumulative functions of time versus number of vehicles. These figures illustrate how forward and backward propagation combined with constraint detection enable traffic flow forecasts.

1. Flow is propagated forward, see Figure 8(a), to calculate a cumulative function at the end of the link.
2. Maximum capacities are incorporated, queues are calculated and propagated backward, resulting in a new cumulative function, see Figure 8(b), at the beginning of the link.
3. The new cumulative function is propagated forward once more to obtain the new cumulative function, see Figure 8(c), at the end of the link.

7 Conclusion and Future Work

In context of the EU FP7 MODUM project, the Traffic Radar is part of a larger infrastructure. On the one hand, real-time traffic data is captured and provided, while on the other hand user requests are provided through various interfaces. The current Traffic Radar platform enables the generation and maintenance of short-term forecasts. This short-term forecasts provide information about the traffic flow and density on links and nodes and can be consulted by both exploring and intention D-MAS. This platform will be tested on a big urban traffic network in the short future. In that stage, the full impact of this novel platform will be measured both in simulation and in a smaller scale in real life.

Acknowledgments. This paper reflects the work funded by European project *MODUM* (FP7-288205, *Models for Optimising Dynamic Urban Mobility*).

References

1. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference architecture for holonic manufacturing systems: Prosa. Computers in Industry 37(3), 255–274 (1998)
2. Daganzo, C.F.: A variational formulation of kinematic waves: Solution methods. Transportation Research Part B: Methodological 39(10), 934–950 (2005)

3. De Swert, K., Valckenaers, P., Saint Germain, B., Verstraete, P., Hadeli, H., Van Brussel, H.: Coordination and control for railroad networks inspired by manufacturing control. In: Proceedings of the IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and its Applications, DIS 2006, pp. 201–206. IEEE Computer Society, Washington, DC (2006)
4. Gentile, G., Meschini, L., Papola, N.: Spillback congestion in dynamic traffic assignment: A macroscopic flow model with time-varying bottlenecks. Transportation Research Part B: Methodological 41(10), 1114–1138 (2007)
5. Hewitt, C.: Viewing control structures as patterns of passing messages. Artificial Intelligence 8(3), 323–364 (1977)
6. Holvoet, T., Weyns, D., Valckenaers, P.: Patterns of delegate mas. In: Proceedings of the 2009 Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems, SASO 2009, pp. 1–9. IEEE Computer Society, Washington, DC (2009)
7. Philips, J., Valckenaers, P., Aertbeliën, E., Van Belle, J., Saint Germain, B., Bruyninckx, H., Van Brussel, H.: PROSA and delegate MAS in robotics. In: Mařík, V., Vrba, P., Leitão, P. (eds.) HoloMAS 2011. LNCS, vol. 6867, pp. 195–204. Springer, Heidelberg (2011)
8. Valckenaers, P., Van Brussel, H.: Holonic manufacturing execution systems. CIRP Annals - Manufacturing Technology 54(1), 427–432 (2005)
9. Van Belle, J., Philips, J., Ali, O., Saint Germain, B., Van Brussel, H., Valckenaers, P.: A service-oriented approach for holonic manufacturing control and beyond. In: Borangiu, T., Thomas, A., Trentesaux, D. (eds.) Service Orientation in Holonic and Multi-Agent Manufacturing Control. SCI, vol. 402, pp. 1–20. Springer, Heidelberg (2012)
10. Van Belle, J., Saint Germain, B., Valckenaers, P., Van Brussel, H., Bahtiar, R., Cattrysse, D.: Intelligent products in the supply chain are merging logistic and manufacturing operations. In: Proceedings of the 18th IFAC World Congress (August 2011)
11. Verstraete, P., Valckenaers, P., Van Brussel, H., Saint Germain, B., Hadeli, K., Van Belle, J.: Towards robust and efficient planning execution. Eng. Appl. Artif. Intell. 21, 304–314 (2008)
12. Woolridge, M.: Introduction to Multiagent Systems. John Wiley & Sons, Inc., New York (2009)
13. Yperman, I.: The Link Transmission Model for Dynamic Network Loading. PhD thesis, KU Leuven (2007)

A Multi-Agent System for Modelling Urban Transport Infrastructure Using Intelligent Traffic Forecasts

Abdallah Namoun[1], César A. Marín[1], Bart Saint Germain[2], Nikolay Mehandjiev[1], and Johan Philips[2]

[1] Manchester Business School, University of Manchester, Manchester, United Kingdom
firstname.lastname@mbs.ac.uk
[2] Department of Mechanical Engineering, KU Leuven, Leuven, Belgium
firstname.lastname@mech.kuleuven.be

Abstract. This paper describes an integrated approach for modeling transport infrastructure and optimising transport in urban areas. It combines the benefits of a multi-agent system, real time traffic information, and traffic forecasts to reduce carbon-dioxide emissions and offer flexible intermodal commuting solutions. In this distributed approach, segments of different modes of transport (e.g. roads, bus/tram routes, bicycle routes, pedestrian paths) are simulated by intelligent transport agents to create a rich multi-layer transport network. Moreover, a user agent enables direct interaction between commuters' mobile devices and the multi-agent system to submit journey requests. The approach capitalises on real-time traffic updates and historical travel patterns, such as CO_2 emissions, vehicles' average speed, and traffic flow, detected from various traffic data sources, and future forecasts of commuting behaviour delivered via a traffic radar to calculate intermodal route solutions whilst considering commuter preferences.

Keywords: Multi-agent system, traffic forecasts, transport infrastructure, path finding, dijkstra algorithm, A* algorithm, intermodal route guidance.

1 Introduction

The transport sector is the second biggest source of greenhouse gas emissions after the energy sector in Europe. It is responsible for approximately 20% of overall greenhouse gas emissions in Europe, with more than two-thirds of that being released from passenger cars. Moreover, CO2 emissions of transport have increased by 23% between 1990 and 2010 [1] despite the introduction of energy-efficient vehicles. To achieve sustainability in transport, there is a pressing need to manage and optimize transport, especially road transport, in a more efficient manner by reducing road congestions and idling times of vehicles, and by offering commuters more eco-friendly forms of commuting such as buses and trams.

Recent research efforts in the field of transportation have focused on optimizing urban traffic and reducing traffic congestion using agent-based computing [7] owing to its distributed nature and ability to deal with uncertainty in an environment [3].

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 175–186, 2013.

Examples include the TRACK-R platform [5] which recommends route solutions using travel time as the cost between any two nodes. TRYSA [6] is another example of a decentralised multi-agent platform for reducing congestion in motorways where agents simulate sections of the motorway. However, none of these efforts aimed at reducing congestion by using a multitude of actual indicators such as CO_2 emissions and vehicles' average speed, and enabling multimodal coordination and cooperation through a multi-modal agent system.

To optimise the management of urban transport, we propose the use of software agents which act as self-organising and self-steering entities to simulate transport infrastructure and traffic conditions and find alternative intermodal commuting routes that satisfy the preferences and requirements of commuters as well as to reduce carbon dioxide emissions. The model proposed herein contributes by, firstly, accounting for several traffic-related variables including ***estimated carbon-dioxide emissions, real-time traffic flows, vehicles average speed, and intelligent traffic forecasts***, and secondly, suggesting ***intermodal route recommendations*** that fit user preferences and demands. The traffic forecasts are generated by a traffic radar which is based on the PROSA [26] and Delegate multi agent architectures [27].

The remaining of this paper is organized as follows. Section two reviews the related literature about traffic management and simulation. Section three proposes our multi-agent traffic management system and details its architecture. Section four discusses the implications of the multi-agent system in respect to managing urban traffic. Finally, section five outlines a future research plan for testing the multi-agent system.

2 Related Work

The transport domain deals mainly with people moving from one place to another in an urban environment. This domain is commonly seen as dynamic because of all the players and their interactions involved in it, for instance private cars, buses, trains and underground among others. Additionally, there is the infrastructure to consider: streets and sensors underneath, traffic lights, and (electronic) signs. Typically, the need for transport is called demand and involves the flow of vehicles, pedestrians and freight, and the transport offer is called supply which involves the infrastructure and services [16].

The fluctuations in the transport demand during a period of time, unexpected accidents and faults, delays, energy consumption, and a limited transport supply make this domain increasingly complex and calls for the need of approaches to distribute the demand within the infrastructure and optimise its usage [16]. Existing approaches focus on diverse and specific aspects of transport, for instance on traffic coordination at traffic management centres, cf. [17], maximisation of (urban) network throughput by controlling the signalling at adjacent road intersections, cf. [18], or improvement of the traffic flow by optimising both the timing of traffic lights and routes selected by drivers, cf. [16]. Even when some approaches have shown improvements, there is still work to do especially to support the ability of incorporating and adapting from sudden transport changes, cf. uncertainty [8].

Software agents and multi-agent systems [15] are increasingly being used to solve overarching issues in the domain of traffic and transportation systems, namely traffic congestions, vehicle emissions, and transportation coordination. The applications of agent computing paradigm ranged from modeling and simulating traffic, managing congestion, and dynamic routing and recommendations [8]. Urban traffic control strategies to reduce traffic jams include intersection signal control [9], bus fleet management [10], integration of urban traffic control and route guidance [12], and intelligent route guidance [11].

Agent-based platforms for road traffic management include TRACK-R for route guidance [5], MAS incident manager for incident management [19], Mobile-C for uncertainty handling [20], and aDAPTS [21] for traffic congestion management. These platforms use agents to model the behavior of vehicles, signal lights, and road segments. Other research efforts focused on modeling the behaviour and reactions of drivers, e.g. upon receiving traffic updates and news [22]. Agent simulation is also applied to the management of bus networks by modeling buses operation and travelers behaviours [23]. The movement of pedestrians in urban environments has also been modeled to help evaluate the effect of infrastructural changes on walking patterns [24]. The majority of these agent-based architectures model only a single mode of transport and do not account for the diverse needs of commuters, nor do they explicitly target transport carbon-dioxide emissions. However, our architecture endeavours to fulfill this gap by simulating all common modes of transport in urban areas, relying on real time traffic data and traffic forecasts, to create a richer and multi-layer transport network that is capable of devising flexible route guidance.

3 A Multi-Agent System for Optimising Transport Using Traffic Forecasts

3.1 Overview of the Multi-Agent System

The agent-based model aims to simulate the transport infrastructure and help end users find appropriate routes for their travel requests, where a travel request contains an origin and a destination. An intelligent agent-based traffic management model should have knowledge about the current status of transport at any time and be able to devise suitable optimizing strategies accordingly. The model we are proposing is advantageous in several ways. For instance, it addresses the issue of multi-modality by covering and combining four modes of transport. In this respect software agents simulate road segments, public transport segments, cycling routes and pedestrian paths (see Fig. 1). Physical properties of such infrastructure are collected from urban and traffic management and control centres (for the cities of Nottingham, UK and Sofia, Bulgaria) and delivered to the multi agent system. Another important functionality of the multi agent system is the calculation of the cost of segments which is then used to find the most environment-friendly routes for a particular user. Cost of each segment is determined through a number of real time variables such as average speed of cars, CO2 emissions for a particular road segment, traffic flow, bus/tram reservations, and travel patterns. These variables are collected and processed separately by the traffic data sensors and SCOOT system [2] of the UTMC centres and updated on a regular basis (e.g. every 2 minutes). Other static information of segments are also taken into account (e.g. physical length of segments, number of lanes, and capacity of public transport).

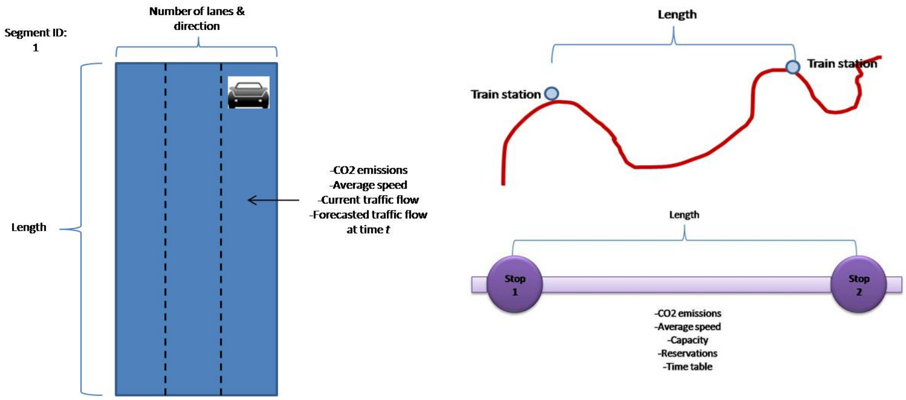


Fig. 1. Road Segment (left), Cycling Segment (top right), & Tram/Bus Segment (bottom right)

Characteristics of our multi-agent system include:

- **Intermodal:** covers a variety of modes of transport, thus accommodating the needs of various types of commuters.
- **Communicates with the commuter:** the multi-agent system receives user journey requests and provides route solutions that satisfy user preferences (e.g. time of travel).
- **Capitalises on a wide range of data sources**, varying from physical properties, real time data, and historic data. These data come from actual traffic management control centres.
- **Simulates a real world problem:** road transport contributes to 20% of the pollution and CO_2 emissions in Europe [1], and is distributed geographically and highly dynamic.
- **Uses real-time data:** agents receive real-time information about traffic from urban traffic control centres and available traffic sensors (e.g. SCOOT data [2]). This enables the agents to recognise the load of the transport network and any traffic problems.
- **Distributes traffic information**: agents communicate their costs to neighbouring agents, thus propagate traffic information throughout the transport network.
- **Exploits traffic forecasts**: our model extends existing models by exploiting forecasts about the situation of traffic in the future via the traffic radar.

3.2 Architecture of the Multi-Agent System

The proposed multi-agent system for providing route guidance contains an environment and a number of intelligent agents living in the environment. We chose to use the AGLOBE framework [4] to design and implement the multi-agent system. Our choice of this agent framework is motivated by the efficiency of message communication and lightweight nature of the framework. The founders of AGLOBE performed a number of benchmark tests against rival agent development platforms JADE, ZEUS, FIPA OS, and JACK and showed the superiority of AGLOBE over these frameworks [4]. Other benefits of using AGLOBE include the ability to simulate real-world problems and collaborative communities, increase the number of autonomous agents in the environment, and flexibility migrate between different containers and platforms.

The components of our multi-agent system were devised depending on the flow of information to and from the commuter on the one hand, and from and to the urban transport centre data sources on the other hand. In principle, a commuter would submit a journey request via a mobile device. The request along with other relevant commuter preferences, such as available modes of transport and preferred time of travel, are collected by a user agent of the multi-agent system. Therefore, the user agent enables the interaction between commuters and our multi agent system, and communicates journey requests to the managing agent, which coordinates and manages the actions of all agents (e.g. sensor agent and transport agents). Information about the transport infrastructure and situation of traffic is collected from available urban traffic management centers using a sensor agent. This information include static properties of transport infrastructure which are requested once at the start of the simulation, such as length of road and public transport segments, number of lanes per segment, and maximum speed for each segment. However, the dynamic properties of traffic and transport are produced and collected every 2 minutes including average speed of vehicles, flow rate of traffic, CO_2 emissions, and reservations and capacity of public transport. These traffic updates are further enriched with forecasts about the situation of traffic at a particular time in the future. These forecasts represent traffic flows, and are generated and maintained by the "traffic radar". Forecasts are periodically sent to the multi-agent traffic system and consist of cumulative flows on links and nodes. Density, speed and flow can be determined for each link/node based on these forecasts for a limited time-horizon (e.g. 2 hours).

The Traffic Radar follows the PROSA architecture [26] and uses the Delegate Multi agent system patterns to coordinate the different agents [27, 28]. Although the in-depth explanation of the "Traffic Radar" is outside the scope of this paper, some properties of the "Traffic Radar" are given below.

- Journeys can be virtually executed considering traffic flows, speeds and densities. The result of a virtual journey gives information on the predicted travel times and speeds on the visited nodes and links. Multiple journey generators finding a route from the origin to the destination can be distinguished:
  - Random search or heuristic search
  - Guided by latest known eco-route which is given by the multi agent traffic system
- Users collect alternative journeys and select a journey as intention. Automatic selection by means of a user avatar (representing the user`s preferences) is also possible.
- Known intentions (time and route) are accumulated on links and nodes in order to predict the link flows, speeds and densities. This information is matched to the historical and the current traffic state.

Effects of unpredicted traffic conditions (e.g. car accident or rain) are also incorporated whenever they occur. The traffic flows are adapted accordingly.

Transport segments within the transport network are modeled using infrastructure/transport agents, which exhibit all collected properties, calculate a cost based on these properties, and communicate with neighbouring agents regardless of their type, e.g. a road agent can communicate to a bicycle agent if they are physically adjacent. This approach creates a multi-layer of transport network, enriched with transport and traffic information. Every time a journey request is solved, the managing agent sends the route solution to the sensor agent, which forwards the solution to the commuter's mobile device. The environment hosts and controls the execution of all agents as depicted in Fig. 2.

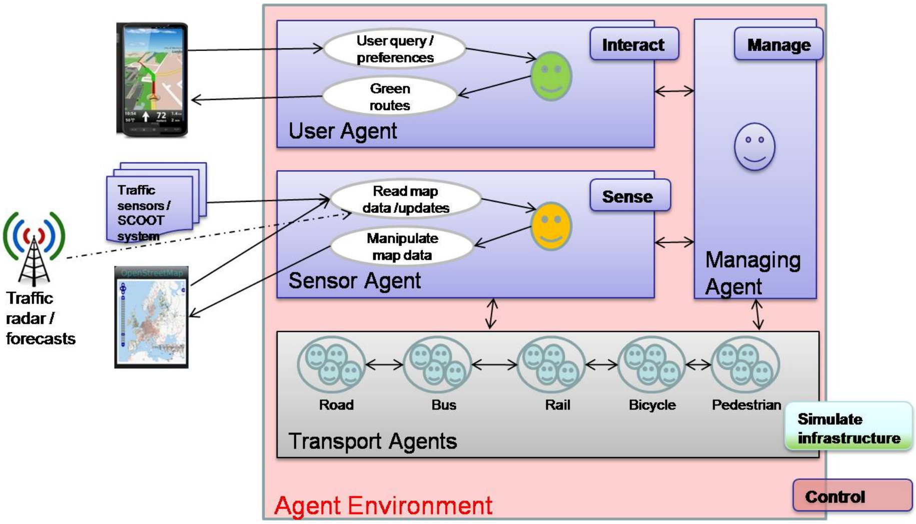


Fig. 2. A Multi-agent Architecture for Modeling Urban Transport Using Traffic Forecasts

3.3 Path Finding in the Multi-layer Transport Network

Path finding in weighted traffic networks is the process of searching for the shortest possible route between two points. The path is represented via a set of segments starting with the origin segment and finishing with the destination segment. Our multi-agent system creates a multi-layer transport network of different modes, where the weight of each transport segment is the cost of using this particular segment. Since our research focuses on reducing carbon dioxide emissions resulting from transport, we selected relevant emission-related variables to calculate the cost of road segments, as opposed to the use of the segment length only as in most typical cases. Therefore, the cost takes into account five main factors: *length of segment, current flow rate of transport, estimated* CO_2 *emissions, average speed of vehicles, and forecasted flow of transport in the future*. Length of segments is obtained from open street map data, whilst other real-time variables are obtained from other data collection systems, e.g. SCOOT [2]. Cost is determined according to function (1):

$$Cost\ of\ transport\ segment = \sum_{k=0}^{n} x_k\ a_k \tag{1}$$

where x is a configurable weight, and a is one of the above five factors, normalised. The weights add up to '1'. It is important to note here that current traffic flow is obtained from actual traffic sensors, while forecasted traffic flow is anticipated by the traffic radar. The above cost function is configurable and allows adding up other traffic variables as needed.

Once a weighted transport network is created for each mode of transport, the multi-agent system can process user journey requests and calculate potential intermodal routes according to the three algorithms below, which are selected rather for general testing purposes of the cost function (1). Yet a performance comparison between these algorithms and other approaches (e.g. ant colony optimisation) as well as a complexity analysis are left out of the scope of this paper.

- **Dijkstra algorithm**: this is a graph search algorithm for finding the shortest path between any two nodes in a positively weighted graph. For a particular source node, the algorithm finds the path with the lowest cost between that node and any other node. This algorithm has been widely used in route planning applications, with variations to improve its performance, e.g. [13].
- **A* algorithm**: this is an improved version of the Dijkstra algorithm, with a reduced search space. It uses a heuristic function to estimate the cost between any segment and the destination segment [14]. In our case, the heuristic function calculates the approximate physical distance between segments and the destination segment which is then used to determine which segment to visit next during the search.
- **Noticeboard algorithm**: this algorithm (details provided in [12]) aims to deliver multi-modal and innovative solutions, with each transportation segment monitoring the travel requests and "volunteering" to take a part in those travel requests which are relevant. A transportation segment can be a street but also could be a train journey or a bus journey. One segment would usually not be sufficient to satisfy the complete transportation request, in which case several segments will form a consortium. Each segment will "price" their participation in the consortium depending on their projected load at the time, where less busy and more eco-sound segments will be cheaper. This will allow a multi-criteria comparison of alternative routes, which will be represented as competing consortia of segments.

In our implementation, we endeavor to pre-process the weighted traffic network prior to finding routes to improve the performance of the aforementioned algorithms. The weights of the multi-layer network are recalculated only when traffic updates or forecasts are received by the sensor agent, otherwise the existing weights are used to recommend routes in which case no unnecessary agents' communication will occur. It is also worth mentioning that as soon as traffic changes occur, the multi-agent system recalculates the routes and dynamically updates the commuter with the new changes.

4 Implications for Traffic Management and Route Guidance

This research capitalises on the advancements, which software agents bring about to solve an increasingly important problem, that of the drastic increase of CO2 emissions as a result of traffic jams. Software agents are intelligent entities that can behave autonomously and react to unexpected, and fast-changing conditions of a specific environment [7]. These very factors quality agents to be an appropriate candidate for handling traffic conditions.

The subsequent table lists the factors that justify the rationale behind our architecture and how it addresses the problems of contemporary traffic in urban cities.

Table 1. Challenges of Urban Transport and Advantages of our Architecture

Main Challenges in the Contemporary Urban Transport	Features of Agents	Our Architecture
Continuous increase of car use and carbon dioxide emissions	**Situated:** direct connection with the environment through sensors and effectors	Models CO_2 emissions through multiple indicators and the cost represents the level of environment-friendliness of each segment
Need for manual interference to guide traffic. For example, human manipulation of phasing lights	**Autonomous**: ability to react independently	Commuters are re-routed automatically in real-time using their mobile devices to avoid major hold-ups and congestion
Traffic disturbances happen unexpectedly (e.g. traffic jam following an accident)	**Reactive**: ability to react to stimulus in the environment in a timely manner	Uses forecasts (e.g. accidents, rush hour … etc) about traffic flow to optimise solutions
Traffic affect each other (e.g. traffic jam on one road could affect traffic flow on another road in the same network region)	**Sociable**: agents talk to each other by exchanging messages	Transport agents communicate to each other regardless of their type (e.g. road, bus, rail) creating a rich and complementary multi-layer network
Lack of synchronisation between differing modes of transportation. A commuter who needs to combine the use of tram and bus, for example, might find it challenging to find a journey that suits his needs. This is due to the lack of integration between varying modes of transport	**Cooperative**: although autonomous in nature, agents can cooperate to achieve the same goal (e.g. the system's goal)	Calculates intermodal solutions that combine various modes of transports to fit commuter demands and constraints

To elaborate the advantages and application of our multi-agent system in route guidance, we discuss a worked example comparing route solutions without and with traffic forecasts. Let us consider a road region from a transport network which consists of 7 interconnected road segments, where ‘O’ signifies the origin and ‘D’ signifies the destination. The nodes in Fig. 3 represent road segments with their IDs. The numbers on top of each node signify the overall cost of using that node, e.g. the cost of using segment 3 is 15. The links in the graph represent the connections between transport segments, for example segment 4 is linked to segment ‘O’ and segment 5.

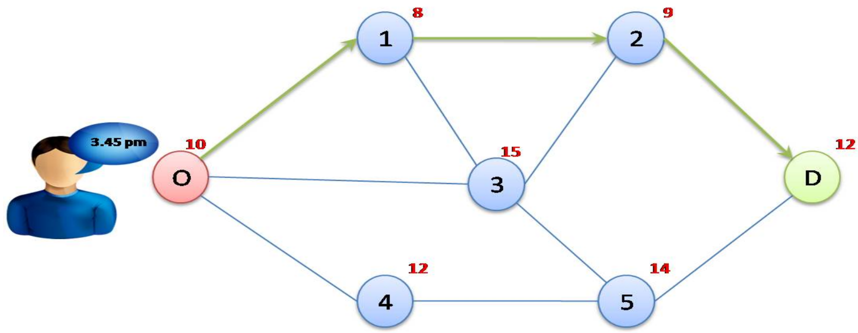


Fig. 3. Case 0, System Routing Recommendations without Traffic Forecasts

In the first case ‘0’, the commuter wants to travel from ‘O’ to ‘D’ at 3.45 pm and submits his journey request using a mobile device. In this case, the user agent is notified with the new request which is then forwarded to the managing agent. The managing agent orders the sensor agent to fetch the latest updates about the current situation of traffic. As soon as available traffic updates are collected from traffic sensors, the road agents update their properties, recalculate their cost, and inform their neighbouring segments. Finally, the multi-agent system calculates a route solution for the commuter request and returns {O, 1, 2, D} to the mobile device via the user agent. The key point in this scenario is that the multi-agent system does not consider the situation of traffic in the future (e.g. at 4 pm) when estimating cost.

Unluckily, the commuter reaches segment 2 at exactly 4pm to find himself stuck in a huge congestion caused by traffic emerging from another road connected to segment 2, with a new cost 47 (Fig. 4). Such route guidance not only aggravates the traffic situation but also damages commuter’s trust in the system.

In case ‘1’, the multi-agent system receives, through the user agent, two new journey requests to travel from ‘O’ to ‘D’ at 3.45 pm and 4pm respectively. This time the road agents use the estimates of traffic flow at 3.45pm and 4pm received from the traffic radar to recalculate their anticipated overall cost. These costs are then updated as indicated by the arrows in Fig. 4. For instance, the cost of segment 1 is anticipated to increase from 8 to 12. For the first journey request, the multi-agent system dynamically diverts the commuter at segment 1 to evade the congestion predicted at segment

2 resulting in a new route solution {O, 1, 3, 5, D}. For the second journey request, the user agent returns {O, 4, 5, D} as the greenest solution to the mobile device. In both situations, our multi-agent system successfully exploits traffic forecasts to evade potential traffic jams and congestions and therefore reduce CO_2 emissions.

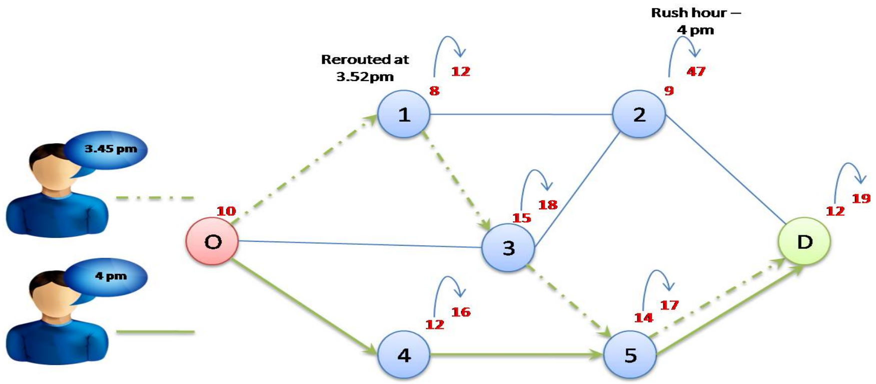


Fig. 4. Case 1, System Routing Recommendations with Traffic Forecasts

5 Conclusions and Future Plan

The paper discusses a multi-agent architecture that uses traffic forecasts to recommend route solutions to commuters. The architecture is innovative in the sense it covers various modes of transport including road, public transport, cycling routes, and pedestrian paths. It also uses various measures to calculate the cost of using each segment. The outputs are intermodal route solutions that accommodate commuters' preferences and demands such as time of travel and possible modes of travel. The multi-agent system is currently being tested using real-time Nottingham city transport data, with 755 actual road segments. It is part of our future research agenda to test and compare the performance of the above three algorithms in the case of serial journey requests and parallel journey requests. To make the comparison even more interesting, we are implementing a centralized and distributed version of each algorithm.

The dynamic updates of traffic to feed our multi agent system come primarily from sources of real time traffic data available in two participating traffic and management centres, Nottingham in the UK and Sofia in Bulgaria. Traffic data sensors in these two cities are indeed different, with Nottingham providing a rich landscape of traffic information about road density, traffic flow, and capacity of roads. Moreover, these two participating sites form two interesting cases to deploy and test our system. It is our goal to evidence that the multi-agent system is applicable to different contexts with substantially diverse capabilities, and able to solve traffic problems relevant to each case. However, we anticipate the multi-agent system to perform better in the Nottingham test site.

Acknowledgement. This research is funded by the FP7 European Project, MODUM.

References

1. `http://ec.europa.eu/clima/policies/transport/vehicles/index_en.htm` (accessed April 14, 2013)
2. `http://www.ukroads.org/webfiles/tal04-95.pdf` (accessed April 15, 2013)
3. Adler, J.L., Blue, V.J.: A cooperative multi-agent transportation management and route guidance system. Transportation Research Part C: Emerging Technologies 10(5-6), 433–454 (2002)
4. Sislak, D., Rehak, M., Pechoucek, M.: A-globe: Multi-Agent Platform with Advanced Simulation and Visualization Support. In: Web Intelligence. IEEE Computer Society (2005)
5. Garcia-Serrano, A.M., Vioque, D.T., Carbone, F., Mendez, V.D.: FIPA-compliant MAS development for road traffic management with a knowledge-based approach: The TRAKC-R agents. In: Proceedings of Challenges Open Agent Systems Workshop, Melbourne, Australia (2003)
6. Hernandez, J.Z., Ossowski, S., Garcia-Serrano, A.: Multiagent architectures for intelligent traffic management systems. Transportation Research Part C: Emerging Technologies 10(5), 473–506 (2002)
7. Wooldridge, M.J.: Introduction to Multiagent Systems. John Wiley & Sons, Inc., New York (2001)
8. Chen, B., Cheng, H.H.: A Review of the Applications of Agent Technology in Traffic and Transportation Systems. IEEE Transactions on Intelligent Transportation Systems 11(2), 485–497 (2010)
9. Roozemond, D.A.: Using intelligent agents for pro-active, real-time urban intersection control. European Journal of Operational Research 131(2), 1, 293–301 (2001)
10. Belmonte, M.V., Pérez-de-la-Cruz, J.L., Triguero, F.: Ontologies and agents for a bus fleet management system. Expert Systems with Applications 34(2), 1351–1365 (2008)
11. Shi, X., Xu, J., Xu, Y., Song, J.: A simulation study on agent-network based route guidance system. Intelligent Transportation Systems, 248–253 (2005)
12. Li, R., Shi, Q.: Study on integration of urban traffic control and route guidance based on multi-agent technology. IEEE Intelligent Transportation Systems 2(12-15), 1740–1744 (2003)
13. Fan, D., Shi, P.: Improvement of Dijkstra's algorithm and its application in route planning. In: Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), vol. 4, pp. 1901–1904 (2010)
14. Cao, W., Shi, H., Zhu, S., Zhu, B.: Application of an Improved A* Algorithm in Route Planning. In: WRI Global Congress on Intelligent Systems, GCIS 2009, vol. 1, pp. 253–257 (2009)
15. Panait, L., Luke, S.: Cooperative multi-agent learning: The state of the art. Autonomous Agents and Multi-Agent Systems 11(3), 387–434 (2005)
16. Bazzan, A.L.C., de Oliveira, D., Klügl, F., Nagel, K.: To adapt or not to adapt – consequences of adapting driver and traffic light agents. In: Tuyls, K., Nowe, A., Guessoum, Z., Kudenko, D. (eds.) Adaptive Agents and Multi-Agent Systems III. LNCS (LNAI), vol. 4865, pp. 1–14. Springer, Heidelberg (2008)
17. Hernandez, J.Z., Ossowski, S., Garcıa-Serrano, A.: Multiagent architectures for intelligent traffic management systems. Transportation Research Part C: Emerging Technologies 10(5-6), 473–506 (2002)

18. Basile, F., Carbone, C., Chiacchio, P., Boel, R.K., Avram, C.C.: A hybrid model for urban traffic control. In: 2004 IEEE International Conference on Systems, Man and Cybernetics, vol. 2, pp. 1795–1800 (2004)
19. Tomás, V.R., García, L.A.: Agent-Based management of non urban road meteorological incidents. In: Pěchouček, M., Petta, P., Varga, L.Z. (eds.) CEEMAS 2005. LNCS (LNAI), vol. 3690, pp. 213–222. Springer, Heidelberg (2005)
20. Chen, B., Cheng, H.H., Palen, J.: Integrating mobile agent technology with multi-agent systems for distributed traffic detection and management systems. Transportation Research Part C: Emerging Technologies 17(1), 1–10 (2009)
21. Wang, F.Y.: Agent-based control for networked traffic management systems. IEEE Intelligent Systems 20(5), 92–96 (2005)
22. Wahle, J., Bazzan, A.L.C., Klugl, F., Schreckenberg, M.: The impact of real-time information in a two-route scenario using agent-based simulation. Transp. Res. Part C: Emerging Technol. 10(5/6), 399–417 (2002)
23. Meignan, D., Simonin, O., Koukam, A.: Simulation and evaluation of urban bus-networks using a multiagent approach. Simul. Model. Pract. Theory 15(6), 659–671 (2007)
24. Kukla, R., Kerridge, J., Willis, A., Hine, J.: PEDFLOW: Development of an autonomous agent model of pedestrian flow. Transp. Res. Rec. 1774, 11–17 (2001)
25. Carpenter, M., Mehandjiev, N.: An Agent based Approach for Balancing Commuter Traffic. In: Proceedings of 19th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE 2010), Larissa, Greece (2010)
26. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference architecture for holonic manufacturing systems: PROSA. Computers in Industry 37(3), 255–274 (1998)
27. Holvoet, T., Valckenaers, P.: Exploiting the Environment for Coordinating Agent Intentions. In: Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.) E4MAS 2006. LNCS (LNAI), vol. 4389, pp. 51–66. Springer, Heidelberg (2007)
28. Verstraete, P., Saint Germain, B., Valckenaers, P., Van Brussel, H., Van Belle, J., Hadeli: Engineering manufacturing control systems using PROSA and delegate MAS. International Journal of Agent-Oriented Software Engineering 2(1), 62–89 (2008)

The Art of Negotiation: Developing Efficient Agent-Based Algorithms for Solving Vehicle Routing Problem with Time Windows

Petr Kalina[1], Jiří Vokřínek[2], and Vladimír Mařík[3]

[1] Intelligent Systems Group, Czech Technical University in Prague, Czech Republic
petr.kalina@fel.cvut.cz
[2] Agent Technology Center, Czech Technical University in Prague, Czech Republic
jiri.vokrinek@fel.cvut.cz
[3] Department of Cybernetics, Czech Technical University in Prague, Czech Republic
vladimir.marik@fel.cvut.cz

Abstract. We present an ongoing effort in developing efficient agent-based algorithms for solving the vehicle routing problem with time windows. An abstract algorithm based on a generic agent decomposition of the problem is introduced featuring a clear separation between the local planning performed by the individual vehicles and the global coordination achieved by negotiation. The semantics of the underlying negotiation process is discussed as well as the alternative local planning strategies used by the individual vehicles. Finally a parallel version of the algorithm is presented based on efficient search diversification and intensification strategies. The presented effort is relevant namely for (i) yielding results significantly improving on all previous agent-based studies, (ii) the inclusion of relevant widely-used benchmarks missing from these studies and (iii) the breadth and depth of the provided evidence and analysis including relevant comparison to the state-of-the-art centralized solvers.

Keywords: multi-agent systems, transportation logistics, optimization.

1 Introduction

The vehicle routing problem with time windows (VRPTW) is one of the most widely studied problems in the the area of operation research (OR), featuring in many real-world logistics or supply chain management applications. It is a problem of finding a minimal set of routes starting and ending at a single depot serving a set of geographically scattered customers, each within a specific time-window and with a specific demand of goods. The VRPTW has been extensively studied for nearly thirty years. Due to the high complexity of the problem itself as well as it's numerous applications the emphasis has been put in developing efficient heuristic algorithms providing good quality solutions in reasonable time, with most successful algorithms being centralized and monolithic e.g. [10,12].

The multi-agent based solving approaches, on the other hand, have not received much attention by the OR community. However, the multi-agent systems

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 187–198, 2013.

are an emerging prominent architecture with respect to modeling new-generation systems based on smart actors and their intelligent coordination, promoting the autonomy of the actors and the clear separation of local decision making and the global cooperation that is beneficial in many real world scenarios. With the shift seen all over the industry towards the internet of things, smart objects and decentralized control systems, this work represents an effort to provide a sound alternative to the *classical*, centralized algorithms. Central to the presented effort is the focus on performance, analysis and relevant comparison to the state-of-the-art classical algorithms missing from previous similar studies.

2 Problem Statement and Notations

Let $\{1..N\}$ represent the set of customers with the depot denoted as 0. For each customer c_i let (e_i, l_i, s_i, d_i) denote the earliest/latest service start times (*time window*), service time and the demand. Let a sequence of customers $\langle c_0, c_1, ..c_m, c_{m+1}\rangle$ denote a single route with c_0 and c_{m+1} denoting the depot. Let D denote the vehicle capacity.

The objective of the VRPTW is finding a minimal set of routes serving all customers. For each route $\langle c_0, c_1, ..c_m, c_{m+1}\rangle$ the sum of corresponding customers' demands must be lower than the capacity of the vehicle serving the route $\sum_1^m d_i \leq D$ (capacity constraint) while the service at each customer c_i must begin within the interval given by (e_i, l_i) (time-windows constraints).

3 Related Work

As mentioned, the VRPTW has been extensively studied for many years. We refer the reader to the excellent surveys of the classical methods [2,3,14] — both exact and approximate — and focus mainly on relevant agent-based studies. The performance of individual algorithms is evaluated using the well known *cumulative number of vehicles* (CVN) metric, corresponding to the total number of routes across all problem instances over the corresponding benchmark.

3.1 Classical Algorithms

For the smaller Solomon's benchmark (100 customers) the best-known overall result was presented by [10] with a CVN of 405. The algorithm is based on the *ejection pools* principle, performing very good potentially unfeasible insertions of customers to individual routes and subsequently recovering the feasibility by ejecting some other customers from the unfeasible routes. The insertion-ejection phase is interleaved with a local search procedure dynamically improving the solution throughout the solving process.

An improved algorithm presented in [12] is based on similar concepts. An ejection-pools mechanism is used accompanied by a powerful insertion method denoted as *squeeze* as well as a search diversification *perturb* procedure. The *squeeze* method also employs a specific adaptive local search procedure used to repair potentially unfeasible intermediate solution using heuristic carried over from the previously mentioned work. The algorithm achieves a CVN of 10290 over the extended Homberger's benchmark (200–1000 customers).

3.2 Agent-Based Algorithms

As mentioned above, none of the previous agent-based studies were particularly successful. The common pitfalls were the lack of relevant comparison to the state-of-the-art algorithms and a generally weak performance.

The algorithm presented in [5] builds on the concepts of a Shipping Company and a Shipping Company Truck. The planning is based on the well known contract net protocol (CNP) accompanied by a *simulated trading* improvement strategy based on [1]. No relevant performance assessment is provided.

The algorithm presented by [9] is based on agents representing the customers, individual routes and a central planner agent. A sequential insertion procedure based on Solomon's I1 heuristic is followed by an improvement phase in which the agents propose moves gathered in a "move pool" with the most advantageous move being performed. Also a route elimination routine is periodically invoked which is not well described in the text. The algorithm achieves a CVN of 436 over the Solomon's benchmark. No relevant runtime information is provided.

In [4] an algorithm is introduced based on Order Agent — Scheduling Agent — Vehicle Agent hierarchy. The algorithm is based on a modified CNP insertion procedure limiting the negotiation to agents whose routes are in proximity of the task being allocated, focusing on the efficiency of the negotiation process rather than solution quality. No relevant performance information is provided.

4 Algorithm for VRPTW Based on Agent Negotiation

This work thus represents a rigorous effort aimed at developing competitive agent-based VRPTW algorithms. An abstract negotiation framework is presented featuring the clear separation between the local planning of individual vehicles and the global coordination achieved by negotiation. The alternative negotiation semantics are discussed as well as the alternative local planning strategies. Finally a parallel algorithm is presented. Only brief overview is provided, referring the reader to our previous works for further details [7,8].

4.1 Abstract Algorithm

Underlying the negotiation based solving process is a fitting agent-based decomposition of the solved problem, featuring a top layer represented by a Task Agent, middle layer represented by an Allocation Agent and a fleet of Vehicle Agents present at the bottom level of the architecture.

Task Agent acts as an interface between the algorithm's computational core and the surrounding infrastructure. It is responsible for registering the tasks and submitting them to the underlying Allocation Agent.

Input: Ordered set of customers C, Fleet of empty vehicles — initial solution σ
Output: Solution σ — complete or partial based on success of the process

```
Procedure negotiate(C, σ)
 1: Init reallocate counters r[c] := 0 for all c ∈ C;
 2: while (exists(c ∈ C), r[c] ≤ reallocationLimit)
 3:   dynamicImprove(σ);
 4:   Select first t ∈ {c ∈ C, r[c] minimal};
 5:   I := {c ∈ σ, costCommit(c, v) is minimal};
 6:   if (I ≠ ∅) then
 7:     Randomly select v ∈ I;
 8:     commit(c, v);
 9:     remove c from C;
10:   else
11:     r[c] := r[c] + 1;
12:   endif
13: endwhile
14: finalImprove(σ);
15: return σ;
```

Fig. 1. The Abstract Global Coordination Process

Allocation Agent instruments the actual solving process by negotiating with the Vehicle Agents. The negotiation is conducted based upon task commitment and decommitment cost estimates provided by the Vehicle Agents.

Vehicle Agent represents an individual vehicle serving a route. It provides the Allocation Agent with the above mentioned inputs. These are computed based on local (private) Vehicle Agent's plan processing.

Figure 1 illustrates the abstract global coordination negotiation process instrumented by the Allocation Agent. In essence it corresponds to a series of *negotiation* interactions between the Allocation Agent and the vehicles represented by the Vehicle Agents. The customers are allocated to individual vehicles based on the commitment cost estimates provided by the vehicles (lines 5–10) computed based upon the particular local planning strategy being used.

Within the *dynamic* or the *final* improvement phases (lines 3, 14) the partial solution being constructed can also be modified in a series of further interactions between the agents in order to e.g. escape local minima, address secondary optimization criteria etc. Thus a particular *negotiation semantics* is given by specifying the exact semantics for the dynamic and final improvement phases.

Both the underlying architecture and the negotiation based coordination process are abstract. On the other hand the particular local planning strategies used by the individual vehicles may reflect the potentially rich semantics of the solved real-world problem e.g. heterogeneity of the fleet, loading constraints, complex optimization objectives etc. We argue that due to the clear separation between the abstract global coordination achieved by negotiation and the local planning strategies used by the individual vehicles the agent-based problem decomposition presents some key advantages for modeling specific real-world environments.

Upon termination, in case there are still some unserved customers (solution σ is not complete), the process can be restarted with a different fleet size or an additional vehicle can be added. Determining a fitting fleet size for the initial solution σ is thus a significant factor affecting the algorithm's efficiency, as is the ordering of the customers within the set C. These two attributes are subject to meta-optimization within the parallel algorithm discussed later in the text.

4.2 Negotiation Semantics

One of the core research objectives behind the presented effort is to explore the possible semantics of the negotiation process and the alternative local planning strategies in an effort to provide an efficient algorithm for the VRPTW.

We considered three general algorithm settings with respect to the used negotiation semantics: (i) *Algorithm-B* baseline setting with neither dynamic nor final improvement phases being employed, (ii) *Algorithm-FI* with only the final improvement being employed and finally (iii) the full fledged *Algorithm-DI* featuring both improvement phases.

Three alternative improvement methods were considered with respect to the *dynamic* and *final* improvement phases, described in detail in [7]. The basic semantics in all three cases is that each vehicle identifies some subset of its customers and each of these customers is then allocated in an auction process to the vehicle with the cheapest commitment cost estimate. We refer to this action as a *reallocation* of a single customer. Thus the three methods differ in the choice of the particular subset of customers to be reallocated within each route.

4.3 Local Planning Strategies

Two particular local planning strategies were considered described in [7]. Both are based on the well known *cheapest insertion* heuristic principle. The *travel time savings* (TTS) heuristic is notoriously known to the OR community. The commitment cost estimates correspond the the increase in travel time caused by the insertion of the corresponding customer to the route. The TTS heuristic thus leverages the spatial aspects of the problem, preferring customers in proximity of the corresponding agent's route irrespective of their time-windows properties. It has been shown [15,11] however, that a strategy exploiting also the temporal relations of the individual deliveries can yield significantly better results.

Thus the *slackness savings heuristic* (SLS) introduces elements to the cost structure based on the constraining effects the insertion of a customer has on the corresponding route schedule. Consider a route with customers having wide time-windows spaced within the route in such a way that the service at each customer starts at the beginning of the corresponding time-window. Then possibly a detour can be made prior to serving any of the customers with the resulting shift in the route schedule not resulting in the unfeasibility of the route. Now imagine adding a customer to the end of the route with a very short time-window. The considered potential detours are no longer feasible. This corresponds to a reduction of the route *slackness* — a situations the SLS heuristics effectively helps to avoid.

4.4 Algorithm Complexity

Given a problem with N customers, the abstract algorithm complexity is

$$O^{vn} + O^{ord} + N \times \left(O^{dyn} + O^{alloc}\right) + O^{fin} \tag{1}$$

where O^{vn} is the complexity of estimating the initial fleet size, O^{ord} being the complexity of reordering the set of customers C prior to the solving process. The O^{dyn} and O^{fin} correspond to the complexities of the *dynamic* and *final* improvement phases while O^{alloc} represents the complexity of the auction process of finding the agent with the best commitment cost estimate.

The complexity of the algorithm thus corresponds to the particular negotiation semantic and local planning strategy being used. In the full *Algorithm-DI* setting the resulting algorithm worst case complexity is $O(N^3)$ in case of the TTS heuristic and $O(N^4)$ for the SLS heuristic [7].

5 Parallel Algorithm with Diversification and Intensification Strategies

As already mentioned, the important factors contributing to the efficiency of the algorithm are (i) the improvement methods to be used within the abstract negotiation process, (ii) the used local planning strategy and (iii) the used initial ordering of the set of customers C.

By analyzing the three discussed general algorithm settings, we discovered that the algorithm is sensitive to the choice of the initial customer ordering in all three cases. The sensitivity was most pronounced with the simpler *Algorithm-B* and *Algorithm-FI* settings and less so with the full *Algorithm-DI* setting. Interestingly, none of the orderings was dominant across all problem instances. To the contrary, each ordering performed well on a different subset of problems instances, suggesting that the instances differ in their nature favoring some particular orderings. Given a specific problem instance we thus found that (i) some orderings perform well for both the simple and the complex algorithm settings, (ii) the best results are most consistently found by the more complex settings over these orderings and (iii) that using these orderings even the simpler settings often return very good results.

Let the term *particular algorithm* denote an algorithm with a specific negotiation semantic and local planning strategy. Given a set of customer orderings Ω and a set of particular algorithms Δ the tuple $[O \in \Omega, A \in \Delta]$ corresponds to a single executable *algorithm instance* within the *algorithm configuration space* $\Omega \times \Delta$. The introduced parallel algorithm is thus based on traversing the *diversified algorithm configuration space* $\Omega' \times \Delta'$. In order to exploit the characteristics of the agent-based algorithm discussed above a specific search diversification and search intensification strategies were introduced. A *diversified set of particular algorithms* Δ' was tailored based on extensive experimentation and tuning of parameters using all three discussed algorithm settings. Likewise a *diversified*

Table 1. Solution quality comparison to the state-of-the-art algorithms

Size	*Classical* [10,12]	*Agents* [9]	*Algorithm-DI*	*Parallel*
100	405	+31 (7.7%)	+24 (5.9%)	+16 (4.0%)
200	694	–	+21 (3.0%)	+12 (1.7%)
400	1380	–	+38 (2.8%)	+29 (2.1%)
600	2065	–	+56 (2.7%)	+43 (2.1%)
800	2734	–	+89 (3.3%)	+71 (2.6%)
1000	3417	–	+115 (3.4%)	+83 (2.4%)
All	10695	–	+343 (3.2%)	+254 (2.4%)

set of orderings Ω' was generated from a set of canonical analytically sound orderings using two specific *ordering diversification operators* introduced in [8].

The parallel algorithm traverses the set Δ' starting with the simplest settings and moves towards the most complex ones. For each particular algorithm $A_i \in \Delta'$ the whole set Ω' is traversed with all the corresponding algorithm instances being executed in parallel. Then, prior to processing the next particular algorithm $A_{i+1} \in \Delta'$, the set Ω' is pruned and some orderings are discarded based on the specific *ordering pruning strategy* being used. Three alternative pruning strategies were introduced — the *BP* (Basic Pruning) strategy, the *CSP* (Minimal Covering Set) strategy and the hybrid *CSP+BP* strategy discussed in [8].

Also importantly, the parallel algorithm also efficiently mitigates the previously identified performance penalty resulting from restarting the negotiation process with an increased fleet size when the resulting solution σ is not complete, having a multiplicative effect on the resulting complexity [7]. This is achieved by (i) executing individual algorithm instances with the initial fleet size always targeting a new best found solution and (ii) terminating them prematurely in case they are not likely to yield such an improvement.

Thus the parallel algorithm uses an efficient *search diversification strategy* based on traversing the diversified configuration space $\Omega' \times \Delta'$ generated using two specific ordering diversification operators and by using alternative negotiation semantics within the abstract algorithm negotiation process. The inherent complexity increase is partially offset by an efficient *search intensification strategy* consisting of ordering pruning, improved restarts strategy and terminating the not promising algorithm instances. For further details refer to [8].

6 Experimental Validation

The experimental evaluation is based on the two widely used benchmarks of Homberger and Solomon [6,15]. Together these benchmarks provide a total of 356 problem instances of 6 different sizes of 100 − 1000 customers featuring 6 instance types differentiated by topology and time windows properties.

Table 2. Runtime comparison to the state-of-the-art algorithms

	Nagata[12]	*Lim*[10]	*Parallel*	
Size	Avg. RT	Avg. RT	Avg. RT	Anytime RT
200	1 min	10 min	10 s	57 ms
400	1 min	20 min	2 min	300 ms
600	1 min	30 min	8 min	2 s
800	1 min	40 min	24 min	7 s
1000	1 min	50 min	54 min	14 s

6.1 Overall Solution Quality Analysis

The comparison to the state-of-the-art algorithms in terms of the primary optimization criteria is presented by Table 1. The results are listed for individual problem instance sizes. The *Classical* [10,12] and the *Agents* [9] columns correspond to the state-of-the-art classical and agent-based algorithms. The *Algorithm-DI* column corresponds to the full *Algorithm-DI* setting combined with the SSL local planning strategy as presented in [7]. The last column corresponds to the parallel algorithm with the *CSP+BP* pruning strategy from [8]. The results correspond to the *cumulative number of vehicles* (CVN) for the second column and the respective absolute and relative error for the remaining columns.

In overall the parallel algorithm achieved a CVN of 10949 over all the benchmark instances, corresponding to a 2.4% relative error with respect to the state-of-the-art centralized algorithms, equalling the best known results for 64% of the problem instances. This represents a significant improvement over all previously presented agent based algorithms.

6.2 Runtime and Convergence Analysis

The comparison in terms of runtime with the state-of-the-art algorithms is presented by Table 2. The listed values correspond to the average runtime for individual instance sizes. The last two columns correspond to the *average composite runtime* — the sum of runtimes of all algorithm instances — and the *average anytime runtime* — the time when the best solution was first found considering a parallel execution of the competing instances within the parallel algorithm.

The results illustrate exceptional anytime convergence of the parallel algorithm, with the time before the best solution is found outperforming even the state-of-the-art solvers. The composite runtime is also competitive. We must note, however, that: (i) compared algorithms outperform presented algorithm in terms of CVN and (ii) are not computationally bound.

6.3 Negotiation Semantics Analysis

The abstract algorithm enables for a number of alternative particular algorithms to be tailored based on the particular negotiation semantics and the local

Table 3. Alternative particular algorithms comparison

Algorithm Setting	Slackness Savings	Travel Time Savings
Algorithm-B	7.6%	23.1%
Algorithm-FI	5.5%	11.1%
Algorithm-DI	3.0%	5.2%

planning strategy being used. To provide an insight into the influence of the used configuration to the resulting solution quality we assessed six relevant particular algorithms based on the three general algorithm settings and the two introduced local planning strategies. The results — corresponding to the relative error in terms of CVN over the 200 customer instances — are listed in Table 3.

With the *Algorithm-B* setting there is no possibility to recover from a potentially bad customer allocations taking place in the early stages of the solving process. For example, an early allocation may render some of the subsequent allocations unfeasible due to the time window or capacity constraints, effectively preventing some parts of the search space from being traversed. The *Algorithm-FI* setting extends the *Algorithm-B* setting by allowing some customer reallocations during the final stage of the allocation process. At this stage, however, the partial solution σ is already tightly constrained and the chance of reallocating a customer within σ is correspondingly small. The full *Algorithm-DI* setting significantly outperforms the *Algorithm-FI* setting. Arguably this is due to the fact that the improvements are performed dynamically throughout the allocation process on smaller and therefore less constrained partial solutions.

In overall, our experiments proved that the number of successful reallocations within the dynamic and the final improvement phases is actually very limited, with the success ratio dropping significantly towards the end of the solving process. This further highlights the fact that it is very difficult to escape local minima once the solution σ gets denser. We argue that this could be effectively addressed by introducing extensions to the negotiation semantics allowing for more complex *trading moves* than simple task reallocations. Our preliminary experiments — not part of this study — suggest that introducing a semantics enabling the vehicles to bid even for such customers that don't fit into the corresponding routes (due to capacity or time-window constraints) but at prices reflecting also the necessary ejection of some other customers from the route might yield interesting opportunities.

6.4 Local Planning Strategies Analysis

Table 3 also illustrates the relative success of the two proposed local planning strategies. The results show that the SLS strategy outperforms the TTS strategy in all examined algorithm settings. The SLS strategy is based on estimating the negative effects of the insertions in terms of reduction to the slackness of the corresponding routes possibly preventing future advantageous detours. Especially when applied iteratively throughout the solving process — within the dynamic

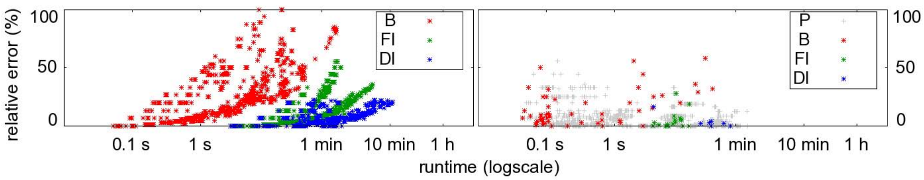


Fig. 2. Illustration of the impact of the introduced intensification strategies

improvement phase in the *Algorithm-DI* setting — the SLS heuristics efficiently prevents the algorithm from being trapped in local minima by modifying the emerging partial solution σ in a way as to increase the overall slackness of the routes and thus increase the chance of further successful allocations.

6.5 Search Diversification and Intensification Strategies Analysis

The highlighted results achieved by the parallel algorithm are based on employing the *search diversification strategy* based on traversing the diversified configuration space $\Omega' \times \Delta'$ generated using two specific ordering diversification operators and by using alternative negotiation semantics. The inherent complexity increase is partially offset by introducing a *search intensification strategy* consisting of ordering pruning, improved restarts strategy and terminating algorithm instances that are not likely to yield good solutions.

Table 1 shows the improvements in solution quality achieved by the introduced search diversification techniques. Compared to the *Algorithm-DI* setting the solutions were improved in 81 cases (23% of problem instances). Both the introduced ordering diversification operators contributed to the improvements in similar way. Our preliminary experiments proved, that the best orderings are found within close neighborhoods of the successful analytically sound orderings, while the orderings featuring greater level of randomization were much less successful. The used operators were thus specifically designed to enable traversing these neighborhoods. An interesting future research opportunity was identified in comparing these operators with well known *ordering crossover* and *mutation* operators used by the genetic ordering based algorithms [13].

The effect of the introduced intensification strategies is illustrated by Figure 2. The two charts capture the runtimes and relative errors of the individual algorithm instances processed by the parallel algorithm over a subset of 16 instances with 1000 customers each. The left chart corresponds to traversing the full configuration space $\Omega' \times \Delta'$ while the right chart corresponds to the intensification strategies being fully employed with the *CSP+BP* ordering pruning strategy being used. The results are grouped based on the used algorithm setting with the terminated algorithm instances being denoted as *P*. Note that the runtime is displayed in logarithmical scale. The results thus outline the dramatic effect the intensification strategies have on the overall runtime of the improved algorithm. The most significant improvements are achieved by (i) terminating instances not potentially yielding good solutions (the P group) and (ii) limiting the number of executed complex algorithm instances by using an ordering pruning strategy (the results missing on the right chart, especially in the most complex *Algorithm-DI* setting group). In overall this attributes to over 6 times reduction in the overall composite runtime without a significant drop in either anytime convergence or overall quality.

7 Conclusion

This paper provides an insight into our ongoing effort aimed at developing efficient algorithms for the VRPTW based on agent negotiation providing a sound alternative to the classical centralized algorithms. Central to this effort is the exploration of alternative negotiations semantics and local planning strategies used within the agent-based solving process.

An abstract algorithm was introduced, based on a fitting agent decomposition of the solved problem. The abstract negotiation based solving process was briefly outlined. The alternative negotiation semantics were discussed based on three general algorithm settings and using three alternative improvement methods based on customer reallocations. Two particular local planning strategies were introduced as well based on the state-of-the-art insertion heuristics. Finally the full parallel algorithm was presented using a search diversification strategy based on traversing the diversified configuration space $\Omega' \times \Delta'$ and a search intensification strategy based on ordering pruning, improved restarts strategy and terminating algorithm instances that are not likely to yield good solutions.

The performance of the algorithm was evaluated using relevant widely used benchmarks providing relevant comparison to the state-of-the-art algorithms missing from previous studies, making this also the first agent-based algorithm to be assessed using the extended Homberger's benchmark. The parallel algorithm was able to equal the contemporary best known solutions achieved by the classical algorithms in 64% of the cases across both benchmarks with an average relative error of 2.4% in terms of the primary optimization criteria, while boasting an excellent parallel anytime characteristics, outperforming even the centralized algorithms in this respect. These results represent a significant improvement over all previously presented agent-based algorithms and suggest that agent-based solving techniques are relevant with respect to efficiently solving the VRPTW, supporting also the relevance of future research in this area.

In that respect, we argue that of particular relevance is the research of the possible semantics of the underlying negotiation process. For example the already promising results could be further improved by introducing a more complex negotiation semantics complementing the simple task reallocations, providing means to modify even the tightly constrained partial solutions and thus effectively escape local minima. Another opportunity was identified in assessing the suitability of known ordering crossover and mutation operators.

Acknowledgements. This effort was supported by the Grant Agency of the CTU in Prague, grant No. SGS12/188/OHK3/3T/13, the Ministry of Education, Youth and Sports of the Czech Republic, grant No. LD12044 and TUD COST Action TU1102 — Autonomic Road Transport Systems.

References

1. Bachem, A., Hochstattler, W., Malich, M.: The simulated trading heuristic for solving vehicle routing problems. Discrete Applied Mathematics 65(1-3), 47–72 (1996); First International Colloquium on Graphs and Optimization
2. Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows, part I route construction and local search algorithms. Transportation Science 39(1), 104–118 (2005)
3. Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows, part II metaheuristics. Transportation Science 39(1), 119–139 (2005)
4. Dan, Z., Cai, L., Zheng, L.: Improved multi-agent system for the vehicle routing problem with time windows. Tsinghua Science Technology 14(3), 407–412 (2009)
5. Fischer, K., Müller, J.P., Pischel, M.: Cooperative transportation scheduling: an application domain for dai. Journal of Applied Artificial Intelligence 10, 1–33 (1995)
6. Homberger, J., Gehring, H.: A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. European Journal of Operational Research 162(1), 220–238 (2005)
7. Kalina, P., Vokřínek, J.: Algorithm for vehicle routing problem with time windows based on agent negotiation. In: Proceedings of the 7th Workshop on Agents in Traffic and Transportation, AAMAS (2012)
8. Kalina, P., Vokřínek, J.: Improved agent based algorithm for vehicle routing problem with time windows using efficient search diversification and pruning strategy. In: Proceedings of the 3rd Workshop on Artificial Intelligence and Logistice (AILog) of the 2012 European Conference on Artificial Intelligence (ECAI), pp. 13–18 (2012)
9. Leong, H.W., Liu, M.: A multi-agent algorithm for vehicle routing problem with time window, pp. 106–111. ACM (2006)
10. Lim, A., Zhang, X.: A two-stage heuristic with ejection pools and generalized ejection chains for the vehicle routing problem with time windows. INFORMS Journal on Computing 19(3), 443–457 (2007)
11. Lu, Q., Dessouky, M.M.: A new insertion-based construction heuristic for solving the pickup and delivery problem with hard time windows. European Journal of Operational Research 175, 672–687 (2005)
12. Nagata, Y., Bräysy, O.: A powerful route minimization heuristic for the vehicle routing problem with time windows. Operations Research Letters 37(5), 333–338 (2009)
13. Poon, P., Carter, J.: Genetic algorithm crossover operators for ordering applications. Computers and Operations Research 22(1), 135–147 (1995)
14. Ropke, S.: Heuristic and exact algorithms for vehicle routing problems, PHD Thesis, Technical University of Denmark (2005)
15. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 35, 254–265 (1987)

Rationales of Holonic Manufacturing Systems in Leather Industry

Sergiu-Adrian Guță[1], Ioan Dumitrache[1], Duncan McFarlane[2], and Tariq Masood[2]

[1] Department of Automatic Control and Systems Engineering, University Politehnica of Bucharest, 313, Spl. Independentei, Sector 6, RO-060042 Bucharest, Romania
idumitrache@ics.pub.ro, guta.sergiu@icpi.ro
[2] Distributed Information & Automation Laboratory, Institute for Manufacturing, University of Cambridge, 17 Charles Babbage Road, Cambridge CB3 0FS, UK
dcm@eng.cam.ac.uk, tm487@cam.ac.uk

Abstract. This paper presents an insight into leather manufacturing processes, depicting peculiarities and challenges faced by leather industry. An analysis of this industry reveals the need for a new approach to optimize the productivity of leather processing operations, ensure consistent quality of leather, mitigate the adverse health effects in tannery workers exposed to chemicals and comply with environmental regulation. Holonic manufacturing systems (HMS) paradigm represent a *bottom-up* distributed approach that provides stability, adaptability, efficient use of resources and a plug and operate functionality to the manufacturing system. A vision of how HMS might operate in a tannery is illustrated presenting the rationales behind its application in this industry.

Keywords: leather industry, leather processing, holonic manufacturing, multi-agent systems.

1 Introduction

Leather has been used since time witness as a luxury material for leather goods, garments, footwear, accessories, upholstery, domestic decorating and furniture. This is due to its numerous advantages over synthetics, namely strength, durability, breathability, unique aesthetic appeal to the eye and hand. Presently leather is made in an estimated 2000 tanneries in the European Union and 9000 worldwide. Technologies employed, standards and size of tanneries vary massively [1, 2]. Tanneries, the plants where leather is processed, are geographically distributed manufacturing systems of high complexity and diversity. Generally in tanneries various equipments such as pumps, valves, motors are placed in areas where cannot be seen or heard by human operators. Thus malfunctions and problems might occur without early detection. In numerous tanneries the activities involved in conversion of the hides and skins into leather are still rudimentary and are not optimized for an aggressive and corrosive environment [3]. Leather manufacturing processes require large quantities of water of precise volume at precise temperature and at the right moment. Chemicals mixed in a variety of formulas, used during leather processing must be dosed properly to

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 199–211, 2013.

maintain consistent quality of finished leather and to reduce negative environmental impact [4]. Nevertheless chemical dosing, water batching, pH adjustment are still done manually [5] leading to operation inefficiencies and exposure to dangerous chemicals.

From economical and technical point of view, in comparison with other industrial sectors, leather processing technologies are seen as being inferior. This view is influenced by the discontinuous mode of operation of production equipments which increases the processing time, pollution and wastes generated by tanneries, consumption of large quantities of water [6] and also the low rate of profit around 10%. Batch processing in leather industry has obvious potential benefits such as flexibility in the assignment of production equipment to multiple products, despite the fact that individual product recipes vary from product to product, possibility of using a variety of raw materials and manufacturing recipes, adaptation to product market demand fluctuations [7]. However, these processes ought to be monitored, controlled and stirred by adequate tools and techniques that ensure a consistent quality of the leather, reduce the waiting time between the operations and increase the productivity of batch operations. Predicting the outcome of any technological change in leather processing is not an exact science and it is also possible to go a long way towards accurate prediction of the effects of concurrent and consecutive processing operations in leather manufacturing [8]. In tanneries the occurrence of unpredictable events is usually unavoidable due to the nature of raw material, complexity of processing operations, customer demand fluctuation, equipment breakdown, and variations in production resources, events that tend to deviate the initial production plan, affect system performance and generate disruptions.

The need for flexible manufacturing systems and control systems which adapt and reconfigure according to the environmental conditions is evident. Although the computer integrated manufacturing concept was expected to deliver the best solution for all manufacturing problems, its implementation resulted in a rigid centralized system, incapable of exhibiting flexibility, robustness, responsiveness and re-configurability [9], whilst essential for any plant to meet current market needs. Holonic manufacturing paradigm that conveys autonomy, cooperation, self-organization and re-configurability to the manufacturing elements promises to fill the gap of these inconveniences. More precisely HMS is a “plug and operate” and “bottom-up” approach that intends to respond to machine breakdowns, introduction of a new equipment/product recipe, order fluctuation in a graceful way providing adaptability and stability to the system. In essence a HMS is formed of holons, autonomous cooperative entities that are in charge of specific identifiable parts of a manufacturing system such as machines, conveyors, dryers, raw materials, pipelines, personnel etc. To illustrate the concept of HMS we present a vision of this approach in leather industry pointing out rationale behind its application.

The paper is structured as follows: section 2 depicts leather manufacturing processes requirements and derives type of the control system needed by tanneries to meet these requirements. Holonic manufacturing concept in leather industry is discussed in section 3. A vision of HMS in a tannery is presented in section 4. Section 5 discusses rationales of HMS in leather industry followed by conclusions in section 6.

2 Leather Manufacturing Requirements and Peculiarities

2.1 Leather Manufacturing Processes

Leather processing involves conversion of putrescible raw hide/skins into a stable material called leather that can be used as a raw material for production of a wide range of goods (shoes, bags etc.). Due to the fact that there is no general or unique procedure for leather processing, the available techniques vary considerably from tannery to tannery are influenced by the nature of the raw material, final product manufactured [10] and also by the experience and talent of tanners. Nonetheless this conversion, which comprises a multitude of different processing operations, up to 70, can be divided in four main phases performed in the vast majority of tanneries: pre-tanning or beam house for cleaning the hides/skins; 2) tanning for stabilization of the collagen matrix; 3) post-tanning to give functional properties; 4) finishing for aesthetics [11, 12]. Processing operations are interlinked determining the overall quality, cost of the final product, and environmental impact. Among these, tanning is an essential operation in which the collagen fibre is permanently stabilized by the cross-linking action on the tanning agents, thus the hide/skin will be immune to micro-organisms and will no longer putrefy or decay. Between 70%-90% of the leather processed globally is tanned using basic chromium sulphate as tanning agent [3, 6] and is unsurpassed in the qualities offered to leather. In addition to mineral tannage based on chrome, vegetable tannage which uses tannins extracted from plants and tannage based on syntans, aldehydes and oils are also used.

Operations carried out in the first three phases of leather processing namely pre-tanning, tanning and post-tanning are often referred to as wet processes, as they are performed in processing vessels filled with water. After post-tanning the leather is dried and operations are referred to as dry processing [13]. Each processing stage requires large quantities of water, approximately 35–40 litres per kilogram (kg) of hide processed [12] at precise volume and at the right time. Along with the water various chemicals in either solid or liquid form are mixed and dosed in a precise way in order to obtain a consistent quality of leather. Water is the medium of transport for chemicals aspect that raises the problem of wastewater treatment. Leather manufacturing processes, the result of each processing phase and also downstream and upstream processes are depicted in Fig. 1.

Characteristics of manufacturing system influence directly the way in which the system operates and is controlled. Manufacturing processes can be divided into two major categories: discrete and continuous. Batch processes pose hybrid characteristics of both categories. “The leather industry falls under the category of jobshop network production facility with specific connotations” [14], more precisely equipments such as drums, paddles or pits can be shared by different final products though their individual recipe might vary from product to product, and each product might follow a different path along the production network.

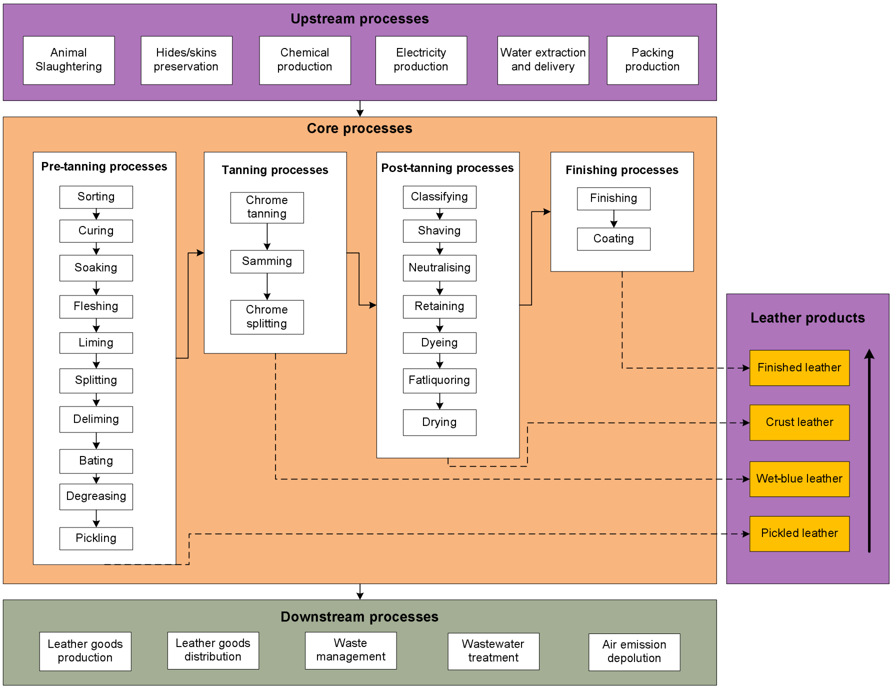


Fig. 1. Main processes of leather manufacturing

2.2 Batch Processing in Leather Industry

Leather processing involves a wide range of chemical reactions and mechanical operations, which are interlinked and carried out mainly in a discontinuous production line flow (batch processing). However drying and finishing operations can be performed in a continuous flow. The raw material, skin or hide has to follow the consecutive sequence of the pre-tanning (beamhouse), tanning, post-tanning and finishing processes without deviation, though some sub-processes can be skipped in order to produce a certain type of leather. Peculiarities and potential of batch processing has been captured by Puigjaner [7] namely: flexibility in assignment of resources to a variety of products, capabilities for processing numerous types of raw materials and /or manufacturing recipes, fast adaptation to fluctuating demand. Batch processing in leather industry is influenced by the following factors [15]:

- Nature and diversity of raw material
- Scale of production
- Variation of market demand
- Frequency of adding or removing products
- Equipment breakdowns
- Reproducibility and complexity of process operations
- Availability and skills of leather technicians

In batch processing in leather industry, skins and hides are processed in groups instead of continuous streams. Hence a specific process is applied to each skin and hide at the same time and a hide or skin does not move to a next stage of processing until the whole batch is done. Tanneries have at least three minimal and necessary components: (i) a market for its products, (ii) a sequence of operations whereby skins and hides are converted into a non putrescible material - leather, (iii) a set of machines in which the operations are performed. In contrast to a continuous plant, where the links between its specific components are well defined at the design stage, the components of a tannery are much less rigidly defined and are subject to change or fuzzy [15]. Leather manufacturing problems could be summed up in the following aspects: (1) financial - production management, (2) process control - process monitoring, product quality evaluation, rework reduction, minimization of waste and empirical practices, (3) social - health and safety, (4) environmental protection - pollution prevention, effluent emission and solid waste reduction, aspects the paper seeks to address.

Although the increasing attention paid to batch processes led to the development of efficient tools for scheduling and planning of process operation, their benefits are limited by the use of hierarchical architecture for their implementation [16], Holonic manufacturing presented in the next section is a paradigm that conveys hierarchical and also heterarchical aspects. Considering leather manufacturing problems, we next deduce a set of properties that the control system of tanneries should have in order to deal with them effectively.

2.3 Main Requirements for the Control System of a Tannery

- **Separation of product recipes from production equipment capability:** allows production/process diversity by keeping recipes procedures separate (Industrial Computer) from production equipment control (Distributed Control System or Programmable Logic Controller PLC). Thus integration of new products/recipes is done easily without changing the code of the equipment controller.
- **Distributed modular architecture:** to enable the scalability of the production by adding or removing production equipments, and also enable easy modification of control scheme.
- **Decentralized control**: a single central controller to deal with complexity of the processing operations performed in tanneries would be inappropriate, a critical point of failure, a source of instability and a bottleneck difficult to change. Decomposing the global system into smaller identifiable parts to which are assigned decisional entities that respond better to locally encountered situations is wiser.
- **Intelligent, adaptive**: in order to respond to changes and disturbances each local decisional entity should posses a certain degree of intelligence to recognize critical situations and take corrective actions by adapting its parameters to the current condition.
- **Reconfigurability**: the control system should be agile enough to quickly re-organize, to make re-configuration of the available resources. The re-configurability should not affect system stability that should continue to function.

3 Multi-Agent and Holonic Manufacturing Concept in Leather Industry

The perspective of an ideal tannery that automatically and dynamically adapts to customer needs, achieves maximum production, efficiency, and also complies with environmental legislation is definitely very attractive to every tannery proprietor. Solving the manufacturing control problem of a tannery can be the key in attaining the mentioned perspective. The manufacturing control can be considered at two levels: (1) low level - process/machine control and (2) high level - system coordination and exploitation. Multi-agent systems (MAS) are generally applied to high level control, but can be used as an enabling technology for holonic manufacturing control which deals with low level control aiming to provide simultaneously real-time responsiveness and greater intelligence to the controlled system.

3.1 Multi-Agent Systems

Derived from (Distributed) Artificial Intelligence, MAS technology is used to solve complex problems that are difficult or even impossible to solve by a single agent or a monolithic system. A MAS is a collection of autonomous entities (software agents) which interact with one another either indirectly via the environment or directly through information exchange. The result of these interactions is an emergent behaviour, a global outcome that surpasses the total benefits of individual agents seen separately. The capabilities of a MAS can be summarized as follows: (i) cooperation in problem solving, (ii) different perspectives of the problem, (iii) experience sharing (iv) modular and flexible development, (v) fault tolerance through redundancy and reuse, (vi) work in parallel for solving common problems. Software agents have no universal definitions [17, 18], however all definitions converge to the fact that an agent is an entity or a software component that has the capability to perform autonomous actions in order to attain its own objectives. In their survey [19] the authors have identified applications of MAS technology in wastewater plant treatment, water flow switching in a network of reactors, energy conservation, control of chemical processes and pH in chemical reactors. The characteristics of MAS makes it a suitable tool and currently the only one for HMS implementations [9] that will be discussed next.

3.2 Holonic Manufacturing Systems

HMS represent a paradigm for addressing most of the major challenges encountered by manufacturing businesses of 21st century. It is based on the concept of *holon,* derived from the Greek root word "holos" meaning whole and suffix "on" meaning a particle or a part as in prot*on,* neutr*on,* which was proposed by Koestler [20]. A holon is an interdependent autonomous entity generally characterized by its classification of unique properties. It consists of a control part and an optional physical processing part. In the literature the terms holon and agent are used in numerous situations interchangeably [9], one difference between them is that the former one is just a concept applied only to manufacturing domain while the latter one is a pure software entity applied to a wider range of domains. The structure formed by holons, which cooperate to deliver the needed manufacturing functionality, is called holarchy. PROSA (Product Resource Order Staff Architecture), the reference architecture for HMS, defines three types of basic holons: order (OH), product (PH) and resource (RH) structured using object-oriented concepts [21]. Characteristics of these holons tailored for leather industry are presented below:

- **Resource holons** are an abstraction of manufacturing resources such as various equipments (tanning drums, dyeing machines), raw materials, leather technicians etc. Each resource available in a tannery is represented by a resource holon. The RHs are able to start processing operations, to monitor and control the process.
- **Product holons** are linked with the product recipes that can be produced in the plant having all the information related to the product recipe such as the quantity of raw materials used (skins, chemicals, water), pH of the float, temperature, processing trajectories (equipments, duration), expected quality of the product.
- **Order holons** represent the manufacturing tasks and can be a customer order, or an order to maintain or repair resources.

  Their capabilities cover main aspects of manufacturing control, products and processes, manufacturing tasks and resources. Staff holons can be added to assist the basic holons with expertise and their global view of the system. ADACOR architecture introduces supervisor holons (SHs), which have a coordination role and provide global optimization in decentralized control [22]- an aspect discussed next.

4 A Vision of Holonic Manufacturing Systems in a Tannery

4.1 Assignment Problem (orders vs. available quality of stock)

Leather manufacturing is a time-consuming process which, in the case of converting raw hides to finished leather, can take at least two weeks. To meet customer demands for products with short life cycle and shorten the time to market, tanners use multiple stocks of intermediary leather products (wet blue, splits, and crust) which can be converted in a shorter time into the products desired by customers. These stocks are usually located in different areas of the tannery or even outside it. MAS can be used to deal with the problem of assignment of available products in stocks to customer orders. Fig. 2. presents a vision in which each stock has an intelligent agent assigned to it, these agents have various information about the stock that they control namely number of batch, origin of the raw skins/hides, date, green weight, wet salted weight and so on. When an order arrives at the sales department, a manager agent is generated. The manager agent is in charge of a customer order and starts to announce the requirements of the order to the stock agents and request bids from those that might meet them. These bids are evaluated according to customer order criteria (article, colour, size, shape) and the winning bidder gets the contract.

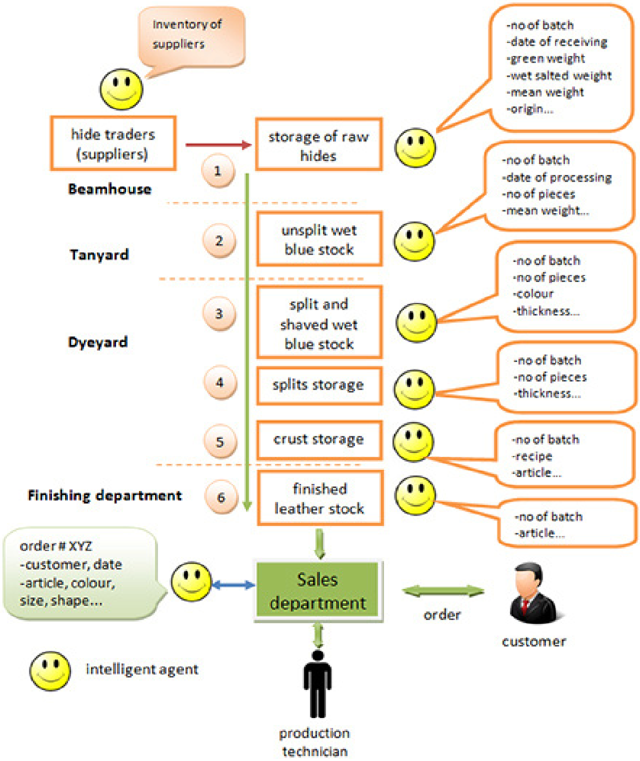


Fig. 2. Stocks in leather manufacturing

In case of partial availability of the required products in the stocks, the manager agent can interact with the production technician who can decide to send a certain batch from a certain stock to further processing in order to obtain the needed quality and quantity of a product. The manager agent can interact with the agent that manages the inventory of suppliers and order more raw materials when needed. The interaction of agents is modelled by Contract Net Protocol (CNP) [23], a widely used coordination protocol inspired by contracting mechanism used by businesses. This approach enables dynamic and flexible assignment of products to customer orders.

4.2 Job-Shop Scheduling

Leather manufacturing is constrained by system conditions (resource capability/availability, processing route, raw material quality), production scheduling problem is considered acyclic. We consider job-shop scheduling as an optimization process that aims to allocate jobs to manufacturing resources in constrained conditions in order to minimize the makespan for a specified horizon time. Traditional approaches to job-shop scheduling includes various methods like analytical, heuristic, metaheuristic (genetic algorithms, Tabu search, simulated annealing), however these methods are inflexible and slow to deal with real industrial situations due to centralized computing [24]. Job shop scheduling using agent technology is a promising approach allowing distributed real-time dynamic scheduling and thus optimization of leather processing operations and a greater shop floor agility.

The processing route of a batch of skins and the machines that perform specific operations is summary illustrated in Fig. 3. A batch can follow a specific route up to a point then is divided in sub-batches that follow different processing route. For example after splitting operation the leather is split in two pieces and the initial batch is divided in two sub-batches that are processed differently.

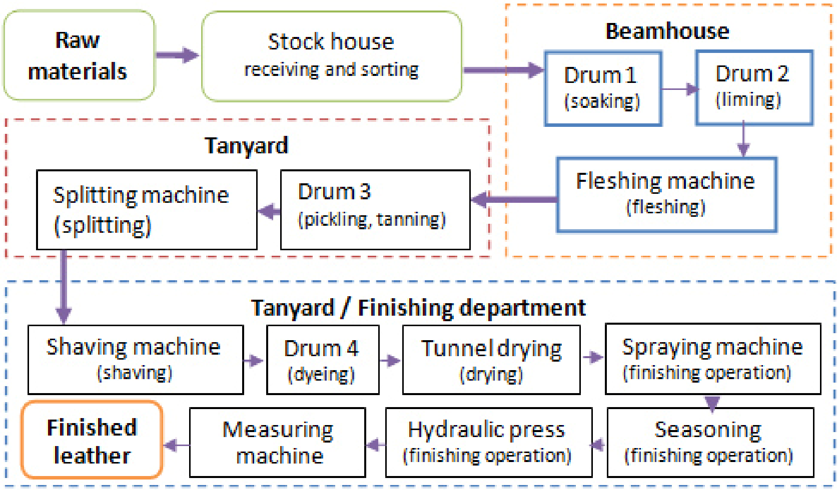


Fig. 3. Machinery layout and flow of a batch of skins/hides from raw materials to finished leather in a typical tannery (summary)

We envisage a distributed approach (Fig. 4.) where functionality of the control system is distributed into four types of holons, namely RHs, OHs, PHs and SHs each of them having a specific role as mentioned (section 3.2). When a production order arrives a set of processing requirements are generated, i.e. processing operations, batch volume, delivery time, leather quality requirements, resources needed which are assigned to OHs. Subsequently the system which was formed by unorganized holons becomes dynamic and organizes itself in order to execute the production order. The OHs communicate their processing demand to RHs, which may have the capabilities to perform them. The OHs try to find resources that best meet manufacturing requirements, while RHs try to maximize their utilization. Through the interaction between OHs and RHs, implemented by employing a negotiation phase, the allocation of manufacturing operations/resources to each production phase is efficiently performed. The SH inspired from [25] is a global observer of the tannery (scheduler), it has a global view of all resources and orders and generates a optimal schedule that is given as advice to the OHs and RHs.

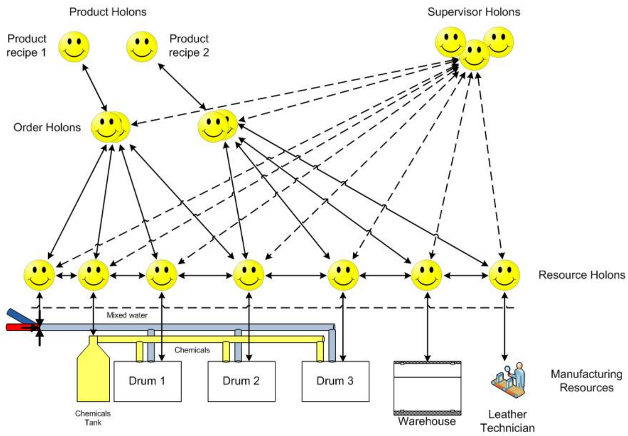


Fig. 4. A vision of holonic manufacturing systems in a tannery

4.3 Process Control

Control of the leather processing operations guarantees consistency in leather quality, ensure leather variety and improve efficiency [5]. Generally automation systems do not offer flexibility and re-configurability due to their centralized control. Resource holons illustrated in Fig. 4., part of a HMS can be implemented to control the hardware of the production equipments including various devices such as pumps, valves, sensors. Implementation of HMS for process/equipment control needs a separation of hardware control into low level control (LLC) for real time control in order to provide real-time responsiveness, high level control (HLC) for hosting the intelligent agents and also an interface for exchanging data between the LLC and HLC levels. In industrial environments LLC and HLC should ideally reside on the same controller like a PLC based on the IEC 61131 standard. In [26] authors presented an agent based approach for batch process automation using two distinct controllers namely a personal computer PC for HLC and distributed controllers for LLC.

In case of an anomaly, the RH linked with a tanning drum tries to fix the problem by asking for example for more water, or chemicals from holons that control them. However, if the RH cannot solve the anomaly, it can ask for assistance from the holon that is linked with the leather technician. On the one hand this vision does not exclude human operators who are supported by holonic and automation technologies via a human-machine interface, but on the other hand they are less exposed to dangerous chemicals and they intervene when is necessary to help with their expertise. The vision presented throughout section 4 is neither exhaustive nor complete, whereas it is aiming to illustrate the idea behind HMS /MAS in leather industry.

5 Rationales of Holonic Manufacturing in Leather Industry

In this section we discuss some aspects related to leather industry challenges and present the rationales behind applying the holonic manufacturing concept in this industry as an alternative to traditional technologies. "Typically, if you add up all of the value-added process steps from raw hide through crust, you get a number that is more or less 72 hours and yet from my experience typically it takes at least 7 days for tanners to move raw hides to crust" [27], this situation is influenced by inefficient resource allocation, use of intermediate storage, spatial distribution of tanneries, but also by the need of resting periods necessary for the completion of manufacturing physical-chemical processes. Holonic distributed approach can be effectively adopted for management of switching between production batches and transportation of intermediate products between different processing equipments. The increased efficiency, in turn will enable mass-customization, since customers can get quick response and the product they want at a lower cost.

Tannery workers have to avoid constant health and safety risks, as well as operational hazards. In a study conducted on a sample of 120 workers, the results revealed that more that 35 % of them showed symptoms of chronic bronchitis, occupational asthma, skin rashes, itching or chronic contact dermatitis [28]. Automatic systems can help with mixing, dosing and addition of chemicals at a certain timetable leading to less exposure to dangerous chemicals and less chemical waste. However, these systems do not provide information about the mutual influences of the processing parameters chemical concentration, pH, temperature, tanning reaction bath length, and do not provide a real process control.

Applying non standard recipes and practices according to one's experience and talent is rather common in E.U. tanneries. This situation is generated by the type variety and by intrinsic variability of hides and skins ranging from light to very heavy, unknown means of preservation, doubtful origin, "blind" process monitoring of closed reactors. In [5], authors argue that control of the leather processing operations guarantees consistency in leather quality, ensure leather variety and improved efficiency. Real process control means not only monitoring processing parameters, as performed by automatic systems, but continuous corrections of the parameters by adjusting chemical concentration with addition of chemicals, by adjusting the pH with addition of acid or alkali, by adjusting temperature with heating or cooling [29]. Implementing intelligent agents at a number of points for process monitoring and controlling that have self adaptive mechanisms to the manufacturing conditions can help minimize rejects, reworking and batch to batch variation.

Analyzing the leather manufacturing requirements, the desired control system for tanneries presented in section 2 and the vision presented in section 4, we can assert that HMS has the potential to effectively address most requirements of leather industry. The requirements are met because of attributes that HMS conveys: autonomy, bottom-up control, reconfigurability, modularity and distribution of control decisions.

6 Conclusion

The paper presented a short synthesis of the leather industry pointing out the peculiarities and requirements of leather manufacturing processes. The need for a new approach to deal with requirements of this industry has been outlined. The desired characteristics of the control system for tanneries were indentified. A simple vision of how HMS might operate in a tannery illustrated the potential of this approach to deal with most of the challenges faced by this industry. Further work will focus on putting into practice this vision in a real tannery.

Acknowledgements. The authors would like to thank Stelian Sergiu Maier from Department of Textiles and Leather Chemistry, "Gheorghe Asachi" Technical University of Iasi, Romania and Ioannis Ioannidis from the Leather and Footwear Institute, Romania for their time and valuable advice in the preparation of this contribution. The first author work was supported by the European Social Fund in Romania, under the responsibility of the Managing Authority for the Sectoral Operational Programme for Human Resources Development 2007-2013 [grant POSDRU/107/1.5/S/76909].

References

1. Nokes, K.: Modern leather manufacturing, `http://www.satra.co.uk/spotlight/article_view.php?id=194`
2. Brugnoli, F.: Social and Environmental Report on the European tanning industry, Bucharest (2012)
3. Thanikaivelan, P., Rao, J.R., Nair, B.U., Ramasami, T.: Recent Trends in Leather Making: Processes, Problems, and Pathways. Critical Reviews in Environmental Science and Technology 35, 37–79 (2005)
4. Hanchevici, A.B., Guta, S.A.: Supervised multi-agent control of leather manufacturing processes by using the fuzzy logic. Leather and Footwear Journal 12, 101–112 (2012)
5. Panda, R.C., Rao, P.G.: Chemical Engineering Applications on Unit Operations of Tanning Industry–Case Studies. International Journal of Chemical Engineering and Applications 2, 112–116 (2011)
6. Deselnicu, V., Badea, N., Maier, S.S., Deselnicu, D.C.: Change through innovation. Part 1. Changing of leathers' tanning system. In: Rusu, C. (ed.) Management of Technological Changes, Book 1, pp. 191–198. Technical Univ. Crete, Liason Office, Univ. Campus, Kounoupidiana, Chania, 73100, Greece (2003)
7. Puigjaner, L., Espuña, A., Santos, G., Graells, M.: Batch Processing in Textile and Leather Industries. In: Reklaitis, G., Sunol, A., Rippin, D.T., Hortaçsu, Ö. (eds.) Batch Processing Systems Engineering, pp. 808–820. Springer, Heidelberg (1996)
8. Covington, A.D.: Prediction in Leather Processing: A Dark Art or a Clear Possibility? Procter Memorial Lecture. Journal of the Society of Leather Technologists and Chemists 95, 231–242 (2011)
9. Babiceanu, R., Chen, F.: Development and applications of holonic manufacturing systems: a survey. Journal of Intelligent Manufacturing, 111–131 (2006)
10. Centre d'Activitats Regionals per a la Producció Neta, Pla d'Acció de la Mediterrània: Pollution Prevention Opportunities in the Tanning Sector Industry Within the Mediterranean Region: Kit. Regional Activity Center for Cleaner Production, Barcelona (2000)
11. Sivakumar, V., Swaminathan, G., Rao, P.G., Muralidharan, C., Mandal, A.B., Ramasami, T.: Use of ultrasound in leather processing industry: effect of sonication on substrate and substances–new insights. Ultrasonics Sonochemistry 17, 1054–1059 (2010)

12. Thanikaivelan, P., Rao, J.R., Nair, B.U., Ramasami, T.: Progress and recent trends in biotechnological methods for leather processing. Trends in Biotechnology 22, 181–188 (2004)
13. European Commission: Integrated pollution prevention and control (IPPC). Reference document on best available techniques for the tanning of hides and skins (2009)
14. Graells, M., Espuña, A., Puigjaner, L.: Optimization of process operations in the leather industry. Computers & Chemical Engineering 16, S221–S228 (1992)
15. Rippin, D.T.: Current Status and Challenges of Batch Processing Systems Engineering. In: Reklaitis, G., Sunol, A., Rippin, D.T., Hortaçsu, Ö. (eds.) Batch Processing Systems Engineering, pp. 1–19. Springer, Heidelberg (1996)
16. Chokshi, N., McFarlane, D.: A distributed architecture for reconfigurable control of continuous process operations. Journal of Intelligent Manufacturing 19, 215–232 (2008)
17. Wooldridge, M.J.: Introduction to Multiagent Systems. John Wiley & Sons, Inc., New York (2001)
18. Bordini, R.H., Wooldridge, M., Hübner, J.F.: Programming Multi-Agent Systems in AgentSpeak using Jason (Wiley Series in Agent Technology). John Wiley & Sons (2007)
19. Metzger, M., Polakow, G.: A Survey on Applications of Agent Technology in Industrial Process Control. IEEE Transactions on Industrial Informatics 7, 570–581 (2011)
20. Bussmann, S., McFarlane, D.C.: Rationales for Holonic Manufacturing Control. In: Proc. of Second Int. Workshop on Intelligent Manufacturing Systems, pp. 177–184 (1999)
21. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference architecture for holonic manufacturing systems: PROSA. Computers in Industry 37, 255–274 (1998)
22. Leitão, P., Restivo, F.: ADACOR: A holonic architecture for agile and adaptive manufacturing control. Computers in Industry 57, 121–130 (2006)
23. Smith, R.G.: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. IEEE Transactions on Computers C-29, 1104–1113 (1980)
24. Shen, W.: Distributed manufacturing scheduling using intelligent agents. IEEE Intelligent Systems 17, 88–94 (2002)
25. Leitão, P.: A holonic disturbance management architecture for flexible manufacturing systems. International Journal of Production Research 49, 1269–1284 (2011)
26. Lepuschitz, W., Koppensteiner, G., Barta, M., Nguyen, T.T., Reinprecht, C.: Implementation of automation agents for batch process automation. In: 2010 IEEE International Conference on Industrial Technology (ICIT), pp. 524–529 (2010)
27. Hottinger, W.: New advances in tannery logistics and automation. Journal of the American Leather Chemists Association 98, 425–430 (2003)
28. Khan, D.A., Mushtaq, S., Khan, F.A., Khan, M.Q.A.: Toxic effects of chromium on tannery workers at Sialkot (Pakistan). Toxicol. Ind. Health (2012)
29. Heidemann, E.: Fundamentals of Leather Manufacture. Eduard Roether (1993)

Adaptive Multi-Agent System for a Washing Machine Production Line

Nelson Rodrigues[1,2], Arnaldo Pereira[1], and Paulo Leitão[1,2]

[1] Polytechnic Institute of Bragança, Campus Sta Apolonia, Apartado 1134, 5301-857 Bragança, Portugal
{nrodrigues,arnaldo,pleitao}@ipb.pt
[2] Artificial Intelligence and Computer Science Laboratory, R. Campo Alegre 102, 4169-007 Porto, Portugal

Abstract. This paper describes the implementation of a multi-agent system in a real industrial washing machine production line aiming to integrate process and quality control, allowing the establishment of feedback control loops to support adaptation facing condition changes. For this purpose, the agent-based solution was implemented using the JADE framework, being the shared knowledge structured using a proper ontology, edited and validated in Protégé and posteriorly integrated in the multi-agent system. The solution was intensively tested using historical real production data and it is now being installed in the real production line. The preliminary results confirm the initial expectations in terms of improvement of process performance and product quality.

Keywords: Multi-agent systems, Manufacturing control, Self-adaptation.

1 Introduction

A current trend in manufacturing domain is the development of adaptive production systems, facing the emergent requirements imposed by global markets demanding high quality customized products at reduced prices, to overcome existing process limitations and enable new manufacturing and processing methods. This challenge, part of the vision for the factory of the future, is also referred in the strategic research agenda made by the Manufuture European Technology Platform [1], which points out the need for enabling technologies, oriented to flexible and intelligent processes that contribute for the achievement of more modular, flexible, re-configurable and responsiveness manufacturing systems.

Multi-Agent Systems (MAS) [2] are a suitable approach to solve these challenges by providing an alternative way to engineer manufacturing control systems, based on the decentralization of the control functions over a set of distributed, autonomous and cooperative entities, the agents. They differ from the conventional centralized, rigid approaches due to their inherent capabilities to adapt to emergence without external intervention [3]. In spite of the promising perspective of using MAS solutions to address the challenge of achieving adaptive production systems, only few industrial applications were reported in the literature, e.g. the application in a factory plant of

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 212–223, 2013.

Daimler Chrysler [4] and some experiences from Rockwell Automation [5] and Schneider Electric [6] (see [7] and [8] for a deeply analysis). Several reasons were identified for this weak adoption by industry, being probably the most important one the convincement of the benefits of using agents running in industry, showing the maturity, flexibility and robustness of the technology.

Under the scope of the GRACE (InteGration of pRocess and quAlity Control using multi-agEnt technology) project (www.grace-project.org), a collaborative MAS solution was developed to operate in an industrial production line, integrating process and quality control, at local and global levels. This approach is aligned with the described trend to build modular, intelligent and distributed control systems, to introduce adaptation facing unexpected deviations and failures, namely in terms of production conditions, product fluctuations and production/process deviations. An important aspect in this work is the achievement of product quality and production performance benefiting from MAS principles, even in a rigid production structure. In fact, MAS are usually useful when the production structure allows alternative ways to re-route the production but in this work the benefits are in terms of product quality and production performance by adapting the process and quality control parameters.

The objective of this paper is to describe the implementation of the GRACE multi-agent system for a production line producing washing machines, and the posterior deployment into real operation. An important contribution of this work is to demonstrate the effective applicability of multi-agent systems in real industrial scenarios, contributing for a wider adoption of this technology by industry.

This paper is organized as follows. Section 2 presents the industrial problem to be addressed and the basic principles of the GRACE multi-agent system architecture. Section 3 describes the implementation of the multi-agent system solution and Section 4 describes how the ontology, supporting the shared knowledge representation, was designed and integrated in the multi-agent system. Section 5 overviews the deployment of the control system into real operation and discusses some preliminary results. Finally, Section 6 rounds up the paper with the conclusions.

2 GRACE Multi-Agent System

The GRACE efforts focus the development of an agent-based system that integrates process and quality control for a production line producing washing machines.

2.1 Description of the Problem

The problem addressed in this work considers a washing machine production line, owned by Whirlpool and located in Naples, Italy (Fig. 1 shows a simplified vision of the line). The production line is composed of several machines arranged in a sequential order, which are linked together by conveyors. Each station performs a single operation in the product being produced, which can be of different types: processing (e.g. bearing insertion or pulley screwing), quality control (e.g. visual inspection or vibration analysis) or manual (e.g. cable and electronics assembly).

Along the line, the quality control stations run proper inspection programs, which results are compiled for posterior analysis. The product instances enter the line with a specific process plan that takes into consideration the materials variables (e.g. type of the rear tub) and the operation parameters (e.g. thickness of welding process) according to the type of washing machine to be manufactured.

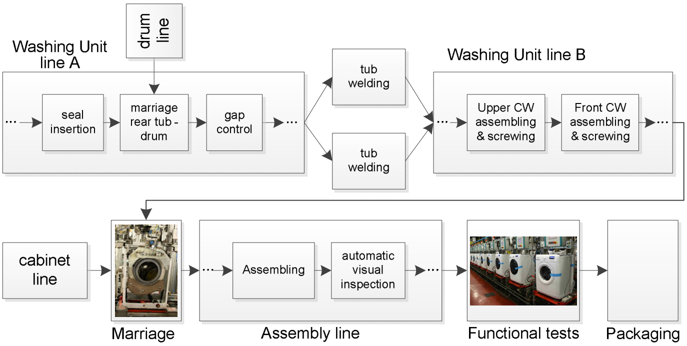


Fig. 1. Layout of the production line case study

The objective to be achieved in this work is the integration of the process and quality control levels, creating feedback control loops that will allow the adaptation of production parameters. Note that the objective is not centred in the re-configurability of the production line (that is fixed and rigid), but instead to introduce adaptation to improve the product quality and the process performance, e.g. reducing the production time, correcting earlier the deviations or quality problems, skipping unnecessary tests along the line and customizing the final product.

An important assumption in this work is to maintain the low-level control, which already uses state-of-the-art industrial control based on Programmable Logic Controllers (PLCs) running IEC 61131-3 control programs, and introduce the multi-agent system solution at a higher control level to provide intelligence and adaptation.

2.2 GRACE Multi-Agent System Architecture

The proposed multi-agent architecture involves a society of distributed, autonomous and cooperative agents representing the components of the production line, to operate at the factory-level. The agents act autonomously on behalf of these components, namely resources or products, introducing intelligence and adaptation, and cooperating to achieve the global production objectives. Several types of agents were identified according to the process to control and to their specialization [9]:

- *Product Type Agents* (PTA), representing the catalogue of products that can be produced in the production line (i.e., different washing machines models). They contain the product and process models.
- Product *Agents* (PA), managing the production of product instances launched in the production line (e.g., washing machines and drums).

- *Resource Agents* (RA), representing the physical equipment disposed along the production line and responsible to manage the execution of their process/quality control operations.
- *Interdependent Meta Agents* (IMA), representing supervision entities that implement global supervisory control and optimized planning.

In the definition of the RAs, several specializations were considered, namely Machine Agents (MA), associated to processing machines, such as robots or screwing stations, and Quality Control Agents (QCA), associated to quality control stations. In these components, two layers were considered:

- A low-level layer, representing the physical machine or testing station. This level may comprise PLCs running IEC 61131-3 control programs.
- A high-level layer, i.e. the agent itself, to perform intelligent management, control and adaptation functions.

The resulting system emerges from the cooperation among distributed, autonomous agents, coordinating their actions along the production line according to the product dependencies and the production plan, as illustrated in Fig. 2.

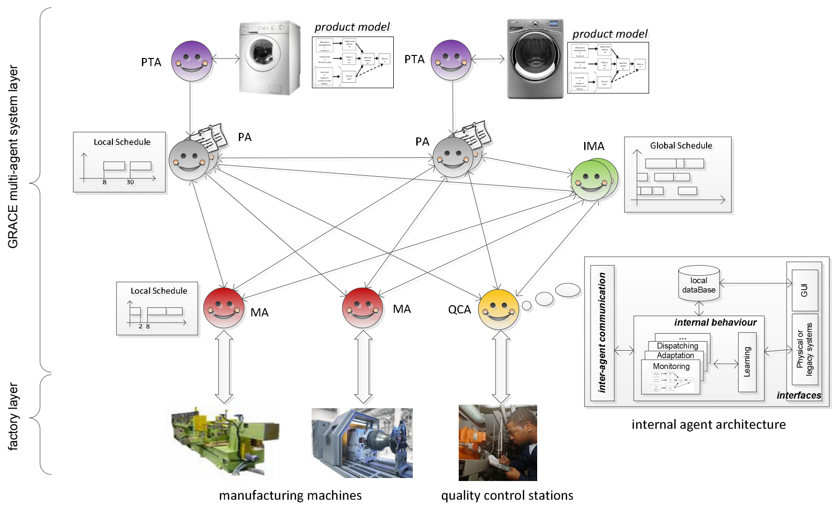


Fig. 2. Multi-agent System Architecture for the Production Line

This interaction among agents is designed to enhance collaboration among intelligent agents providing a sound perspective to achieve:

- Dynamic and run-time adaptation to respond to condition changes, such as unplanned fluctuations of process/product parameters, at local and global level.
- Supervision and control schemes at factory level which maximize the efficiency of production and product quality, through feed-back control loops based on a continuous flow of information among agents.

For this purpose, individual agents exhibit a structure of behaviours according to their individual roles, enhanced by proper adaptation mechanisms, focusing local and global self-adaptation (see [10] for more details).

3 Implementation of the GRACE Multi-Agent System

The development of multi-agent system solutions is strongly simplified if a proper agent development platform is used, taking advantage of the useful services it provides, such as registry and management services. In this work, the Java Agent Development Framework (JADE) [11] was used, since it better responds to several requirements, namely being an open source platform and compliant with Foundation for Intelligent Physical Agents (FIPA) specifications, providing low programming effort and features to support the management of agent-based solutions and delivering an easy integration with other tools, namely the Java Expert System Shell (JESS).

Briefly, JADE is a Java based architecture that uses the Remote Method Invocation (RMI) to support the creation of distributed Java based applications. JADE provides a framework containing several agents that support the management of agent-based applications, namely the Agent Management System (AMS) and Remote Management Agent (RMA).

3.1 Implementation of Individual GRACE Agents

The GRACE multi-agent system comprises four types of agents, the PTA, PA, RA and IMA. Each one of these GRACE agent types is a simple Java class that extends the *Agent* class provided by the JADE framework, inheriting basic functionalities, such as registration services and capabilities to send/receive agent communication language (ACL) messages [11]. These functionalities were extended with features that represent the specific behaviour of the agent, as detailed in [9].

Each GRACE agent is developed using multi-threaded programming constructs, over the concept of the JADE's behaviour, allowing the execution of several actions in parallel. The execution cycle of a GRACE agent is briefly illustrated in Fig. 3.

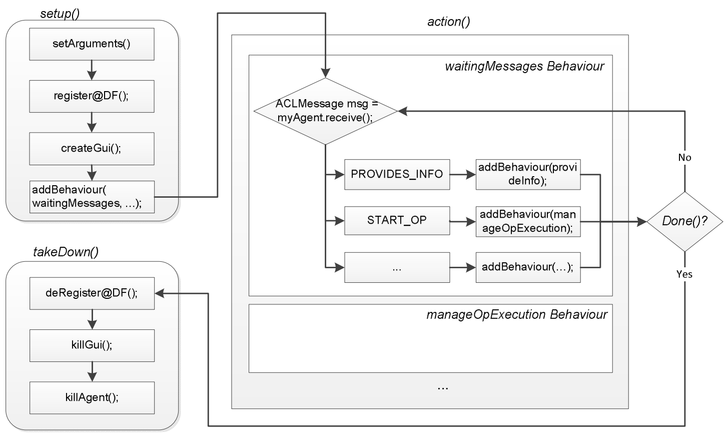


Fig. 3. Execution life-cycle of a GRACE's agent

When the agent is created, the first method to be executed is the *setup()* method that should be fulfilled with the actions to be performed during the agent's start-up. In the GRACE's agents, the initialization procedure is responsible to register the agents' skills in the Directory Facilitator (DF), connect to the local database and create the GUI component. At the end of the *setup()* method, some behaviours are triggered, passing the control to the *action()* method that implements the code related to the desired behaviours. The behaviours launched in the *setup()* method and those posteriorly invoked within these behaviours are provided in a software package in the form of Java classes.

The communication between distributed agents is asynchronous and done over Ethernet network using the TCP/IP protocol. The messages exchanged by the agents are encoded using the FIPA-ACL communication language, being their content formatted according to the FIPA-SL0 language. The meaning of the message content is standardized according to the GRACE ontology.

Since the behaviour of the agent is mainly driven by the messages received from other agents, a cyclic behaviour called *WaitingMessages* is launched in the *setup()* method. This behaviour is a Java class that is waiting for the arrival of messages, using the *block()* method to block the behaviour until a certain time elapses or a message arrives, and the *receive()* method to extract the incoming message. The arrival of a message triggers a set of actions related to decode the message and select the proper behaviours to be performed. As an example, if the received message has a PROVIDES_INFO identifier, a new behaviour called *ProvidesInfo()* is triggered. Note that after triggering the action related to the received message, the *WaitingMessages* behaviour remains continuously waiting for incoming messages.

At the end of the GRACE agent life-cycle, two methods are invoked: i) *done()* that tests if the behaviours included in the action method are finished, and ii) *takeDown()* that performs the last actions of the agent life-cycle, e.g. deregister from the DF.

3.2 Integration of Adaptation Functions

The functions providing the adaptation capabilities were embedded in the agents according to their roles. For this purpose, these functions were aggregated in software packages according to the agents where they will be hosted, being the functions that are shared by more than one agent type grouped in one separate package.

At the PA level, the *configuresOperationParameters()* functions adjust the operation parameters to be executed by the station, considering the data related to previous executed operations. In the case of testing operations, it can involve the selection of inspection algorithms and parameters; in case of processing operations, the selection of components, programs and parameters. Two examples are the customization of the testing plan to be performed by the functional tests station and the adjustment of the parameters to be written in the machine's controller board.

At the RA level, the *adjustParameters()* set of functions adjusts the processing or testing parameters, just before to trigger the execution of the operation, according to the local knowledge of the agent and the information gathered from previous quality control operations. As an example, QCA may use the feedback deriving by the past visual inspections to optimize its parameters and adapt itself to the environmental conditions (e.g. adapting the exposure time of the camera for image acquisition).

At the IMA level, global optimization procedures were deployed to elaborate optimization over identified meaningful correlations, supporting the adaptation of global policies for the system. As an example, the *trendAnalysis()* function performs a short analysis of the collected data to detect deviations or patterns in the data (e.g. a degradation process towards the defect) and generate warnings for other agents.

3.3 Integration with Legacy Systems

An important issue is the integration of the multi-agent system in a computational ecosystem, which is already running in the production line at different control levels. In this work, the integration of legacy systems will be exemplified with the integration of the quality control applications. As referred, in the GRACE multi-agent architecture each quality control station, which can be a physical equipment or an operator, has associated a QCA agent that is responsible to introduce intelligence and adaptation in the execution of the inspection operations.

In this work, several quality control stations are developed in LabView™ [12], imposing specific constraints to be integrated with the Java applications, i.e. the agents. In order to enable the communication, it was necessary to choose among different technological approaches. The best approach is to use a kind of service-oriented approach, where the services encapsulate the application's functionalities. These services can be provided (and announced in a service registry) by the Java application (i.e. the agent) or the LabView™ application (i.e. the quality control station). Since the integration is focused in two pre-defined applications (which doesn't require the need to discover the available services in the distributed system), only a basic layer is implemented, using TCP/IP sockets to interconnect the two applications. This approach allows the easy interconnection of the two applications running in different machines, achieving a very portable solution.

3.4 User Interfaces

Graphical User Interfaces (GUIs) are one way to provide a user interface supporting the management and monitoring of the system. Each type of agent provides a different GUI, since each one handles a particular set of information and allows different types of interactions with the users. In spite of providing different information, the GUIs follow a common template of menus, customized according to the agent's particularities.

The use of a Java based framework to develop the multi-agent system, offers the possibility of using *Swing*, a well-established toolkit to implement GUIs for desktop applications. Each type of agent in the GRACE system has its GUI implemented as an extension of the *javax.swing.JFrame* component. The GUI displays the local information stored in the agent's database, which feed the appropriate graphical components that are relevant to the users. As example, the IMA's GUI provides a global perspective of the entire production system, illustrating what is being produced (actual status) and what was produced (historical data).

4 Ontology for the Shared Knowledge Representation

Ontologies plays a crucial role in the GRACE multi-agent system to enable a common understanding among the agents when they are communicating, namely to understand the messages at the syntactic level (to extract the content correctly) and at the semantic level (to acquire the exchanged knowledge). For this purpose, a proper ontology was designed to represent the knowledge associated to the domain of washing machine production lines, formalizing the concepts, the predicates (relation between the concepts), the terms (attributes of each concept), and the meaning of each term. The ontology schema (see [13] for more details) was edited and validated using the Protégé framework (protege.stanford.edu/), which is an ontology editor and knowledge-based framework.

The integration of the GRACE ontology in the GRACE multi-agent system requires the derivation of the ontological terms from the interfaces and the generation of a set of Java classes. For this purpose, two different approaches can be considered: a manual or an automatic one. In the first, the concepts are derived manually, which is a long, very difficult and time consuming task. The second approach uses tools that automatically translate the ontological concepts into the Java classes.

In this work, the *OntologyBeanGenerator* plug-in was used, which allows to automatically generate the Java classes from the Protégé tool, following the FIPA specifications. The main generated class represents the vocabulary and main ontological objects (i.e. concepts and predicates) defined in the ontology. The second group of generated Java classes specify the structure and semantics of each ontological object defined in the ontology. The methods defined in each individual class, e.g. the getters and setters, allow to handle the data related to the object.

At the end, some hand-made corrections in the exported Java classes are required due to syntax errors introduced by the plug-in during the automatic generation process. However, the use of this plug-in accelerates the integration process of the ontology, being a good approach to manipulate, integrate and reuse ontologies.

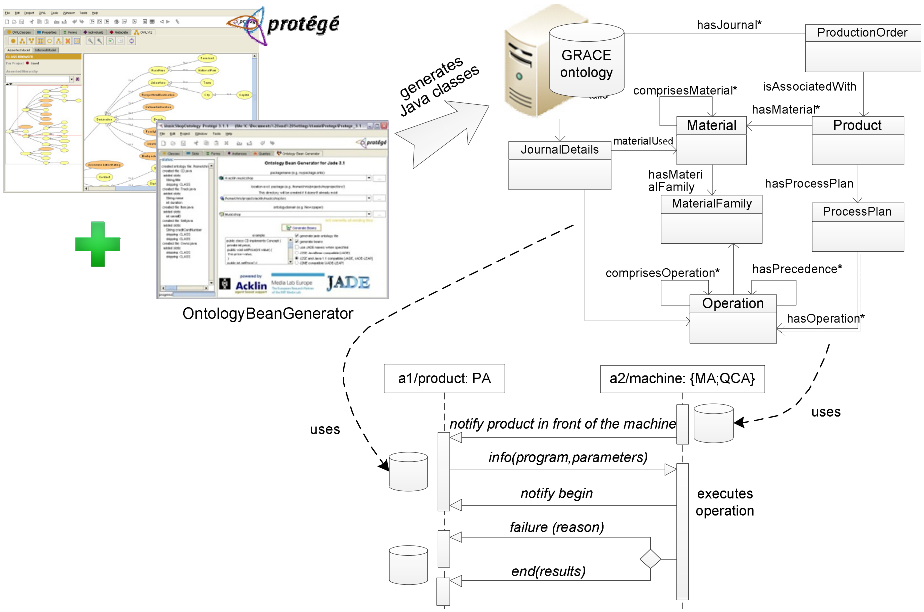


Fig. 4. Agents using ontologies to exchange knowledge

At this stage, all classes needed for the ontological model are created, and ready to be used by the GRACE agents (after registering the ontology). As illustrated in Fig. 4, the agents use the same ontology (but different fragments of the ontology) to express the shared knowledge that is exchanged through the message exchange according to proper interaction patterns following the FIPA protocols.

A local database is used by each agent to store the data handled by the agent, according to the ontology used to represent the GRACE knowledge. To potentiate the solution portability, a SQLite database (http://www.sqlite.org/) was implemented using the SQLite JDBC Driver (https://bitbucket.org/xerial/sqlite-jdbc).

5 Deployment in the Production Line

The developed GRACE multi-agent system was initially debugged by using the unit test approach through the JUnit framework [14] to test each agent's function individually. After this testing phase, the system is ready to be deployed in the production line. This section discusses the agentification of the production line and the analysis of the results from the multi-agent system operation.

5.1 Agentification of the Production Line

The launching into operation of the multi-agent system requires two important actions: launching the JADE platform and launching instantiations of the agent classes developed for the specific agent-based solution, in this case for the Whirlpool's production line producing washing machines.

For this purpose, the structure of the GRACE agent classes described in the previous sections (i.e. PTA, PA, RA and IMA) is instantiated according to the production line needs. The set of agents launched to handle the washing machine production line are distributed by several computers, running in Windows 7 (64-bit) operating system with an Intel(R) Xeon (R) CPU W3565 processor @ 3.20 GHz. In terms of resources, 17 RA agents are launched, being the QCAs installed in the computers where the LabView™ applications for the associated quality control station are running. In another side, 9 PTA agents are launched corresponding to 9 different washing machine models, and 1 IMA is deployed to supervise the production line activities. The number of PAs running in the system is variable, depending on the number of products that are being produced in the production line, but in a stable production flow more than 200 PAs are simultaneously running.

The instances of the agent classes need the customization of their behaviours. For example, each one of the stations disposed along the line will be associated to instances of the RA agent, but each one has its particularities and it is necessary to reflect them in the generic structure of the RA agent. For this purpose, each agent has associated a XML file, describing the particularities and skills of the station. The following example illustrates the XML for the bearing insertion station.

```
<machine>
     <name> Bearing_insertion </name>
     <type> processing </type>
     <id> 5100 </id>
     <line> WU_line_A </line>
     <port> n.a. </port>
     <sqlserver> 159.154.64.164 </sqlserver>
     …
</machine>
```

These XML files are read when the agents initiate their life-cycle, within the *setup()* method, loading the agent profile with the customized parameters.

5.2 Analysis of Preliminary Results

The GRACE multi-agent system was deployed in the real production line. In the first phase, the operation of the multi-agent system was tested using real historic production data (avoiding the need to connect to the physical devices in the factory plant). This stage was crucial to identify and correct possible bugs before proceeding to the deployment into the on-line production of the factory plant. For this purpose, virtual resources were created to emulate the functioning of the physical devices using the production data stored in the production database. Since the database is managed by a Microsoft SQL Server application, the Java Database Connectivity (JDBC) API (Application Programming Interface) is used to establish the connection between the agents and the SQL-based database.

The intensive operation of the GRACE multi-agent system (running non-stop during slots of 1 week), showed, in a first instance, the correctness of the agent-based system, namely its stability and robustness, which allowed to conclude that the multi-agent system is sound and ready to be used in the real production line. Fig. 5 illustrates the multi-agent system running in practice, being possible to verify that the IMA agent had executed a trend analysis and generate a warning to the agent that may be responsible for the detected deviation.

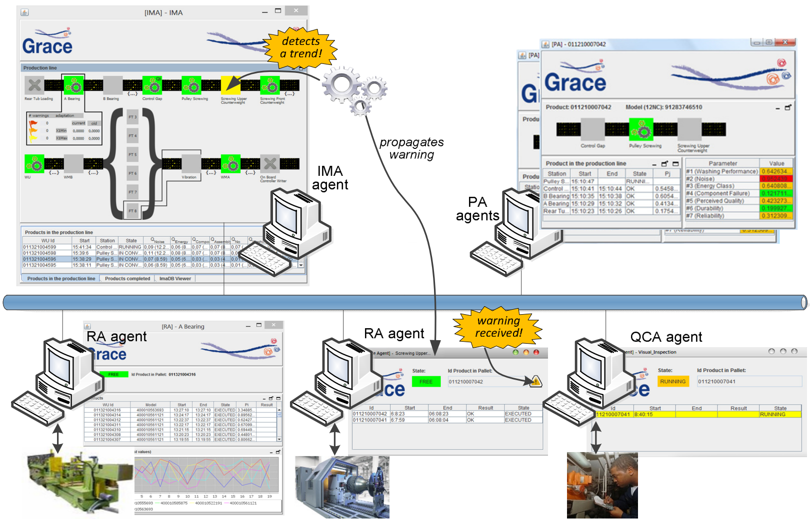


Fig. 5. GRACE multi-agent system working in practice

In terms of quantitative results, besides the improvement of the product quality, it was noticed a reduction of the production time. Particularly, the implementation of adaptation mechanisms in the selection of the functional tests, by removing unnecessary tests, adjusting others or customizing the messages to the operators, allows an increase of:

- Productivity, since the inspection time is reduced (the average reduction of 1 minute over the default 6 minutes corresponds to an increase of approximately 20% in the production line productivity [15]).
- Product quality, since most effective quality control procedures are performed.

The proposed multi-agent system is now being installed in the real production line and the very preliminary results confirm the initial expectations in terms of improvement of process performance and product quality. In fact, after running the GRACE multi-agent system in practice, the average time to execute the functional tests was reduced, but more important was the customization of the messages displayed to the operator that allowed performing the inspection with better accuracy for particular situations. Additionally, the adaptation of operations parameters according to the production history and current environmental conditions, e.g. the customization of the on-board controller, allows a significant improvement of the product quality, since using the proposed multi-agent system, each washing machine is customized according to its production process historic.

The success of the multi-agent system deployment in a real industrial application is a crucial step to prove the applicability and merits of GRACE multi-agent system for production lines, and also the benefits of agent-based control approaches in industry.

6 Conclusions and Future Work

This paper describes the implementation of a multi-agent system in a production line producing washing machines, aiming to integrate the process and quality control and to provide adaptation capabilities to the system operation.

This agent-based system was implemented using the JADE framework, which provides an integrated environment for the development of such systems. The skeleton of the several GRACE agents were implemented, namely the behaviours' structure of each agent, the ontology schema for the knowledge representation, the interaction patterns supported by FIPA protocols, and the integration with legacy systems, particularly the LabView™ applications running in quality control stations. Several GUIs were also implemented to support an easy interaction with the users. Local and global adaptation functions were embedded in the several developed agents, aiming to provide adaptation and optimization based on the integration of the quality and process control.

The multi-agent system was intensively tested using historical real production data, aiming to test and correct the detected mistakes and bugs during the development process. The intensive tests showed the correctness of the system, namely in terms of adaptation, robustness and scalability. At the moment, the GRACE multi-agent system is being deployed in the factory plant and future work is devoted to the complete commissioning of the agent-based solution in the factory plant and the continuously monitoring of the system operation to extract the achieved benefits.

Acknowledgments. This work has been financed (or partly financed) by the EU Commission, within the research contract GRACE coordinated by Univ. Politecnica delle Marche and having partners SINTEF, AEA srl, Instituto Politecnico de Bragança, Whirlpool Europe srl, Siemens AG.

References

1. European Commission: MANUFUTURE, Strategic Research Agenda: Assuring the Future of Manufacturing in Europe, Report of the High-level Group, Brussels (2006)
2. Ferber, J.: Multi-Agent Systems, An Introduction to Distributed Artificial Intelligence. Addison-Wesley (1999)
3. Wooldridge, M.: An Introduction to Multi-Agent Systems. John Wiley & Sons (2002)
4. Schild, K., Bussmann, S.: Self-Organization in Manufacturing Operations. Communications of the ACM 50(12), 74–79 (2007)
5. Vrba, P., Tichý, P., Mařík, V., Hall, K., Staron, R., Maturana, F., Kadera, P.: Rockwell Automation's Holonic and Multi-agent Control Systems Compendium. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 41(1), 14–30 (2011)
6. Schoop, R., Neubert, R., Colombo, A.W.: A Multiagent-based Distributed Control Platform for Industrial Flexible Production Systems. In: Proceedings of the IEEE Int. Annual Conference on Industrial Electronics (IECON 2001), vol. 1, pp. 279–284 (2001)
7. Pechoucek, M., Mařík, V.: Industrial Deployment of Multi-agent Technologies: Review and Selected Case Studies. Autonomous Agents and Multi-agent Systems 17(13), 397–431 (2008)
8. Leitão, P.: Agent-based Distributed Manufacturing Control: A State-of-the-art Survey. Engineering Applications of Artificial Intelligence 22(7), 979–991 (2009)
9. Leitão, P., Rodrigues, N.: Multi-agent system for on-demand production integrating production and quality control. In: Mařík, V., Vrba, P., Leitão, P. (eds.) HoloMAS 2011. LNCS, vol. 6867, pp. 84–93. Springer, Heidelberg (2011)
10. Leitão, P., Rodrigues, N.: Modelling and Validating the Multi-agent System Behaviour for a Washing Machine Production Line. In: Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE 2012), Hangzhou, China, May 28-31, pp. 1203–1208 (2012)
11. Bellifemine, F., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE. Wiley (2007)
12. Johnson, G.W.: LabVIEW Graphical Programming: Practical Applications in Instrumentation and Control, 2nd edn. McGraw-Hill School Education Group (1997)
13. Leitão, P., Rodrigues, N., Turrin, C., Pagani, A., Petrali, P.: GRACE Ontology Integrating Process and Quality Control. In: Proceedings of the 38th Annual Conference of the IEEE Industrial Electronics Society (IECON 2012), Montreal, Canada, pp. 4328–4333 (2012)
14. Beck, K., Gamma, E.: Test Infected: Programmers Love Writing Tests. Java Report 3(7), 37–50 (1998)
15. Fernandes, A.: Simulação de Linha de Produção usando a Plataforma ARENA, MSc Thesis in Informatics Engineering, Polytechnic Institute of Bragança, Portugal (2012)

Product Intelligence in Warehouse Management: A Case Study

Vaggelis Giannikas[1], Wenrong Lu[1], Duncan McFarlane[1], and James Hyde[2]

[1] Institute for Manufacturing, University of Cambridge, CB3 0FS, Cambridge, UK
{eg366,wl296,dcm}@eng.cam.ac.uk
[2] James & James Fulfilment Ltd., CB22 6SA, Cambridge, UK
jh@doublejames.com

Abstract. The need for more flexible, adaptable and customer-oriented warehouse operations has been increasingly identified as an important issue by today's warehouse companies due to the rapidly changing preferences of the customers that use their services. Motivated by manufacturing and other logistics operations, in this paper we argue on the potential application of product intelligence in warehouse operations as an approach that can help warehouse companies address these issues. We discuss the opportunities of such an approach using a real example of a third-party-logistics warehouse company and we present the benefits it can bring in their warehouse management systems.

Keywords: product intelligence, warehouse management systems, adaptive storing, dynamic picking.

1 Introduction

An approach that can treat different product instances in a special way based on their specific characteristics and needs has been argued to bring special benefits both in manufacturing and in supply chain industrial contexts [1]. Focussing on supply chain and logistics operations, the impact of such a product intelligence approach [1, 2] has been recently under consideration in a number of different areas such as road-based logistics [3], intermodal transportation [4] and production logistics [5]. In this paper, we discuss its applicability and potential benefits in another area of logistics operations, other than transportation-related ones —the operations run in warehouses.

Due to the rapidly changing preferences of customers, orders received by warehouse companies (especially third-party-logistics ones) increasingly exhibit special characteristics, such as smaller order size, higher product variety, request of shorter response time, and request for changes after the order has been initially created and placed [6]. This means that although the traditional performance targets for warehouse services (e.g. warehouse utilisation, tighter inventory control) still remain, in today's environment, they are subject to the specific, special needs of different customers. This is particularly true in third-party-logistics

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 224–235, 2013.

warehouses that manage a high variety of products and a big number of individual customers. Here, the operations are required to become more customer-oriented and more responsive to requests with different characteristics and needs in an efficient manner.

In this paper we aim to demonstrate the way the product intelligence paradigm can respond to the above challenges by studying its future application in a third-party-logistics warehouse company. After reviewing the current situation in warehouse management systems in Section 2, we discuss the opportunities for the adoption of intelligent products in warehouse operations and their potential applications in Section 3. We present our scoping case study along with two specific application examples in Section 4 before concluding with our findings.

2 Current Issues in Warehouse Management Systems

The main operations that almost every warehouse needs to plan and control are receiving inbound items from suppliers, storing the items, receiving orders from customers, retrieving the requested items and assembling the orders for outbound shipment, and shipping the completed orders to customers [7]. In order to achieve higher performance for warehousing regarding capacity and throughput, and fulfil the service at the minimum resource cost, warehouse resources (such as space, labour, and equipment) need to be carefully chosen, operated, and coordinated. Therefore, a Warehouse Management System (WMS) becomes essential since it provides, stores, and reports the necessary information to efficiently manage the flow of products within a warehouse [8]. Currently, in order for any standard WMS to be able to overcome today's challenges and retain its competitiveness, two are the main characteristics it should possess: a) flexibility in terms of being responsive to short-term changes of customer demands in a timely manner and b) adaptability in terms of being able to maintain the service level when mid-term changes/requirements are demanded by customers.

Although a WMS should be more responsive to changes in order to enhance its flexibility and adaptability, conventional paper- or spreadsheet-based WMSs are incapable of providing timely and accurate warehouse operation information since they rely heavily on staff members to enter information manually or through a barcode system [9]. On one hand, researchers are trying to solve the issues of timeliness and accuracy on warehouse operations by capturing real-time information using Auto-ID systems (such as RFID) [9, 11] and wireless sensor networks technologies [10]. On the other hand, even though the information technologies mentioned above can provide more accurate and real-time information regarding operations, unexpected events and disruptions, the important challenge for warehouse managers to make decisions using this information in a short response still remains [11]. On this direction, a significant amount of research has focused on developing decision support models (including heuristics and algorithms) which aim to optimally manage different warehouse operations [7]. However, due to their rigid assumptions and constraints, successful implementations of these models in current commercial WMSs are rare [12].

At the same time, most of these models use centralised, static, off-line methods, which raise significant barriers to the development of more flexible and adaptable WMSs.

Similar issues have been recently appeared in manufacturing control and supply chain management where alternative approaches such as the product intelligence one have been developed for their solution [13, 14]. We will discuss the potential application of product intelligence in warehouse management and the opportunities for its successful deployment in the systems managing them in the next section.

3 Intelligent Products in Warehouse Management Systems

Although there are numerous examples in the literature of deployments that use multi-agent approaches in warehouse management systems [15–17], the application of the product intelligence paradigm in warehouse management systems (WMS) and the benefits in the operations they manage is yet to be studied. However, the potential of a distributed intelligence approach, such as a product intelligence one, in WMSs cannot be underestimated due to reasons that might cause centralised management systems not to perform in a efficient way [18]:

- *Partial information availability*: Each possible decision-making node has only part of the information required to make the decision due to the high levels of uncertainty that many warehouse operations face. Examples of such information can be the arrival of new orders during working shifts, the arrival of new pallets/products from the supplier/client during the day, and the real-time location of the pickers in the warehouse.
- *Impracticality*: Even though there are cases where the required information is available to each decision-making node, practical constraints such as time and cost inhibit a centrally based solution. For instance, the optimal picking lists and routes for each and every picker cannot be re-calculated every time a new product or a new order arrives in the warehouse since this process will require a significant amount of time, thus making the previous solution ineffective.
- *Inadvisability*: Even if the above issues do not apply to certain systems, a centralised system might still be inadvisable due to the susceptibility of a single decision-making node to disruptions and changes. Another reason for the inadvisability of the deployment and adoption of a centralised system can be the complexity of making changes driven by new needs and requirements such as new products, capacity levels, facilities layout.

Apart from the aforementioned reasons, which can make centralised approaches in WMSs inefficient, there is also a number of opportunities for the deployment of intelligent products in the management of warehouse operations. Tables 1 and 2 aim to summarise the main areas where an intelligent product approach could be beneficial. We distinguish between opportunities for the application of Level-1 (information oriented) and Level-2 (decision oriented) Intelligent Products [1] in Table 1 and 2 respectively.

In Table 1, it is obvious that the main impact of Level-1 intelligent products refers to the automation of some time-consuming and labour-intensive tasks as well as the collection and usage of product information in static algorithms used for the determination of storage and picking policies/decisions.

Table 1. Opportunities for Level-1 Intelligent Products in warehouse operations

Operation	Opportunities
Receiving	– Better accuracy of received products when products are "checked-in" – Faster scanning of big pallets especially when wireless technologies are being used (e.g. RFID) – Improved visibility of inventory in warehouse facilities even though products are not yet stored in shelves
Storing	– Easier identification of empty shelves that meet product's physical dimensions – Determination of proper storage zones based on real historical data – Faster identification of a product's predefined storage location in a warehouse according to the storing policy – Richer information availability —such as product's turnover rate, demand, picking frequency— for storage location assignment algorithms
Picking	– Identification of products' locations in the warehouse for the determination of the fastest route – Consideration of multiple feasible locations for each product type if several product instances are available – Scheduling subject to trolley's capacity
Shipping	– Identification of proper packaging option for each order – Determination of best delivery service available – Automatic generation of shipping documents

The potential benefits from the automation of certain tasks are also depicted in Table 2 via the utilisation of products capable of deciding how they will be stored, picked and shipped. Moreover, this level of intelligence can facilitate the management of more dynamic scenarios, where unexpected events, changes and disruptions take place in the normal operations of a warehouse. Finally, the role of the customer in some of the decisions made becomes more active compared to the traditional passive role the customer normally has [1].

4 Scoping Case Studies

4.1 Company Background

The company used as a case study for this paper is a an eCommerce and mail order fulfilment warehouse. The company warehouses, picks, packs and dispatches

Table 2. Opportunities for Level-2 Intelligent Products in warehouse operations

Operation	Application areas
Receiving	– Automated proof of delivery sent to supplier – Faster checking of inbound orders' contents against product list – Notification to pickers about the availability of products requested that were previously out-of-stock
Storing	– Guidance to staff members regarding product's storage location in the warehouse – Usage of adaptive storage location assignment approaches: each product instance can be stored in a different location – Negotiation among products and/or with shelves about choosing a storage location
Picking	– Faster problem notification in out-of-stock cases – Dynamic update of picking lists after new orders arrive, orders' status change etc. – Negotiation among products and/or with pickers about updating a picking list
Shipping	– Identification of proper packaging station for each picking list – Choice of preferable delivery service by the client and/or end-customer

goods to end-customers on behalf of their clients, the retailers[1]. Orders can come from a range of channels including online eCommerce stores and marketplaces such as eBay, Amazon and Play and are automatically retrieved by the company via secure cloud servers; the company also has facilities to handle mail and telephone orders. Figure 1 summarises the main operations followed every time a new order is added in the system. Although not depicted directly in Fig. 1, products are normally sent to the warehouse company from their clients before any order is placed from end-customers, that is the warehouse needs to store any products their clients ask them to store (based for example on demand forecasts).

4.2 Challenges and Opportunities for the Case Company

The specific business case offers a number of opportunities for the successful application of product intelligence in its operations:

1. *Available infrastructure*: The Company has already in place an information system capable of uniquely identifying the different products and orders in the warehouse (elements of the information system are presented in Fig. 2). At the same time, the information system in use collects and stores data related to the lifecycle of a product/order (e.g. physical dimensions of a

[1] By the "client" we refer to the warehouse's clients (the retailers) and by the "end-customer" we refer to the customers who put their orders at the retailer's website.

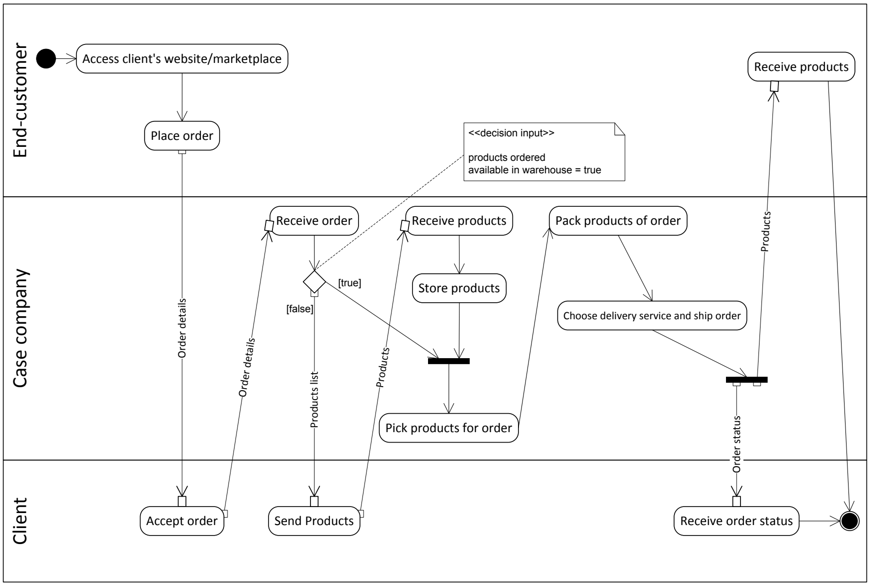


Fig. 1. Case company's main operations

product, demand, location in the warehouse) as well as a customer's needs and preferences associated with his orders (e.g. delivery date, priority, delivery method). In other words, the Company has already developed and used elements of Level-1 intelligence.

2. *Quickly changing environment*: The Company is expanding quickly and unpredictably, meaning that a centralised system able to cope with the challenges faced today might not be suitable for the needs of the Company in the near future. Agreements with new clients (of different sizes) are happening in a regular base bringing new products in the warehouse with their own special information as well as bringing new end-customers that the Company has to prepare the orders for. Moreover, the capacity of the warehouse itself is growing introducing new storage locations, aisles and packing areas.
3. *High customer impact on warehouse's operations*: The specific business model that the Company is using, requires a certain level of control of their operations to be given to their clients and end-customers. For example, the Company might not know when the products from their clients will arrive during the day or what products exactly will the shipments contain. Also, each client might hold information that could be useful for the warehouse's operations like for example weekly/monthly demand, seasonality of products, new seasonal products.
4. *Need for dynamic decisions*: Many of the decisions that need to be made each day could be taken using a centralised, static system, but when multiplied up to thousands of activities and products the identification of the best one becomes a very hard to be solved problem. Static systems will also not

Fig. 2. Technology Infrastructure: Auto-ID technologies for shelves (left) and Personal tablets for staff members (right)

perform very well since they are not capable of utilising real-time information during everyday operations even if this information is gathered and stored. A simple example here are end-customer orders since, in the case company, they are received continuously throughout the day.

As shown in Tables 1 and 2, there are many application areas that a product intelligence approach could be beneficial at. In the next two sections we deep into two specific examples to demonstrate how potential developments would operate in these areas, focussing on cases where the decision making properties of intelligent products could be used.

4.3 Example 1: Adaptive Storage Location Selection

The first example comes from the potential impact of product intelligence on storage policies, that is the selection of the proper storage locations for the incoming products usually aiming at minimising the picking time of future orders. The company in our case study is currently using rule-of-thumb policies [7] choosing between the random location policy and the closest open location one based on simple zoning rules. This is happening mainly because the company cannot know in advance the specific products (and their quantities) they will receive each day from their clients as well as the point of the day they will receive them at. As a result, the usage of static algorithms that determine new storage locations for incoming product instances before the beginning of each working day is simply not practical. At the same time, the company believes that although zoning the warehouse could reduce the overall picking time, these zones will need to change very often (since the demand they are facing is volatile), thus creating confusion to the staff members.

Since the infrastructure is already in place and the staff members are used to be advised by simple interfaces regarding where to pick the products from (see Fig. 2), the company considers a similar solution for the storing process where an automated system will guide the staff members around the warehouse every time they will scan a product that needs to be stored. The location of each product can be different among product instances and throughout time. We call this policy an adaptive storage assignment policy [6], in which each product instance can be treated differently each time it has to be stored in the warehouse. In other words, each product instance (or group), represented by a product agent, will try to seek the best location in the warehouse every time it is received using information such as its turnover rate, demand, the relationships with other products, the layout of the warehouse etc. The optimality of this decision is calculated based on the expected picking time of future orders received by the company. In practice, the staff members will pick some products from the pallets received, load them on their trolleys, scan them and then receive guidelines regarding where to store each of them on the monitor attached to their trolley.

The decision regarding the storage location of the products in an intelligent product approach like the one described above can be made via a number of different ways as shown in Fig. 3. The first option (Fig. 3a) depicts a pure intelligent product approach where the products are the only decision makers. Each product will choose its preferable storage location and in case two (or more) products choose to be stored in the same location, they will negotiate with each other before they reach a decision. On the other hand, in the second option shown in Fig. 3b, the resources (here the storage locations-shelves) get a more active role by accepting or rejecting offers when products ask to be stored in them. This architecture still represents an intelligent product approach since it is up to each product to choose where and how much they will bid on, however, the resources will make the final decision.

It is obvious that both architectures can facilitate the development of an adaptive solution although there are still some open questions (which are out of the scope of this paper) such as how the products choose their preferable locations, what the negotiation mechanism between them is and how the bidding system between products and shelves operates. Another issue that comes from the practical application of such a solution refers to the common practice of storing multiple product instances of the same product type in the same storage location every time a new pallet of products is to be stored. This issue is tackled in Fig. 3c where several product instances form teams and let their "Group Leader Agent" do either the negotiation with other similar agents (Option a) or bid for specific shelves (Option b).

4.4 Example 2: Dynamic Order-Picking Rescheduling

Order-picking operations have been identified as the most labour-intensive and costly activities for almost every warehouse [19]. In such operations, multiple pickers are assigned the task of transferring products from stationary storage locations to a common loading shed or depot, wherein the pickers begin and end the trips in the depot [16]. The case company is currently using two simple rules to operate order-picking. Orders are assigned to pickers based on the orders' priorities (higher priority orders will be picked first) and subject to the capacity of their picking trolley. Then the pickers move in the warehouse linearly, skipping any aisles where there are no picks required and trying to minimise their total walking distance while collecting the requested products. Currently,

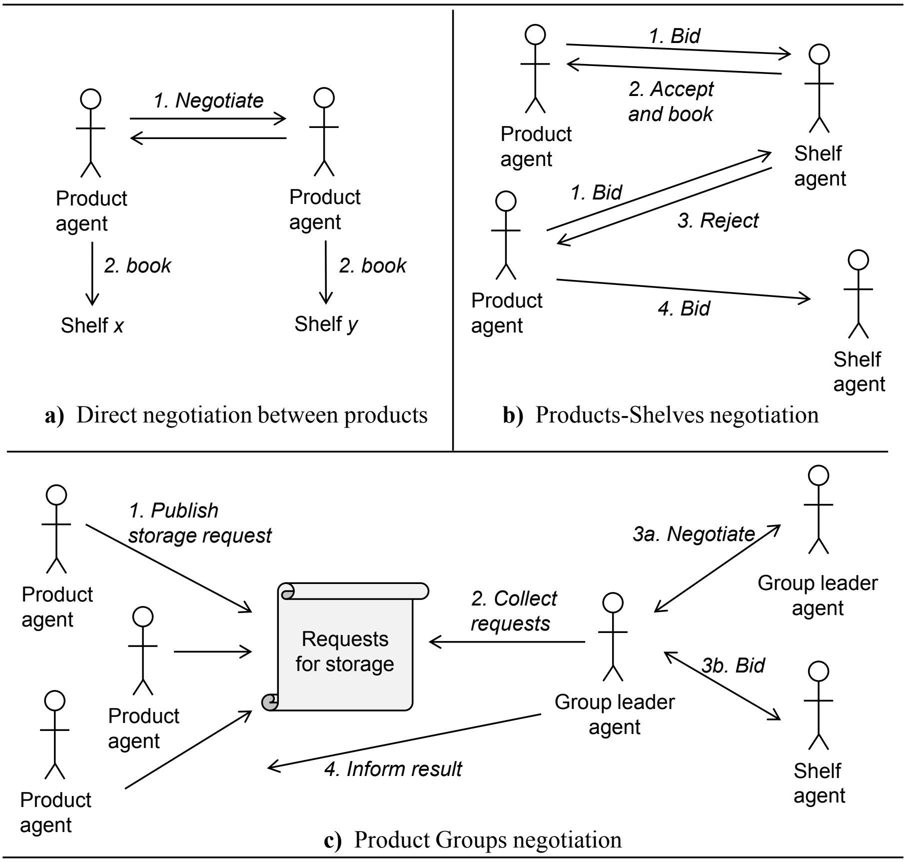


Fig. 3. Agent architectures for the adaptive storage location selection problem

the above policy allows the company to perform in a very good level, however, they understand that it will not be sustainable for the near future. Moreover, critical decision making criteria that affect the performance of the order-picking operation —such as picking sequencing and routing, selection among multiple locations for a requested product type— have been left out from the current order assignment system.

Although there are a number of algorithms for the efficient scheduling of order-picking operations considering the critical criteria mentioned above, most of them are traditionally calculated off-line and consider the problem as being a static and deterministic one [17]. Therefore, the calculated schedule will no longer be optimal in operating environments that continuously change due, for example, to the receipt of new orders at random points during a working day. Hence, an alternative objective is the minimisation of the picking time each time an unexpected event (such as the arrivals of new orders) takes place. Although such a dynamic and automated order-picking rescheduling policy does not guarantee the calculation of a global optimum, it can lead to schedules that do take into account unexpected events and disruptions. In a WMS that uses a product intelligence approach, the products will have the opportunity to communicate with each other and perhaps make a different decision for the final picking lists when something changes in the operation.

We will demonstrate the way a product intelligence approach can operate through a simple example. Figure 4 depicts a warehouse with one depot and six pallet racks, each of them being able to store products in both sides. In this example Pickers 1 and 2 (currently located as shown in the figure) with Picker 1 scheduled to pick the products located in the shelves marked with the star sign (*) and Picker 2 scheduled to pick the products located in the shelves marked with the percent sign (%). Assuming that each picker has a capacity of 10 items, let Picker 1 have two spare spaces in his trolley after he completes his picking task. Now, let a new order, which requires two products enter the system. The available locations that contain the required product are marked with “N”. With the current system (and any other off-line tool), the products for the new order will be picked after a picker is ready to start a new picking task.

In a product intelligence approach, when the new order arrives, all product instances that match the requested product type will flag themselves, indicating that they are requested for another order. According to the status of the pickers, these flagged product instances will inform Picker 1 that he should pick the “N” products before going back to the depot since he still has two spare places in his trolley. In a more intelligent solution the “N” products will identify that Picker 2 is closer to them than Picker 1. However, due to capacity limitations, the picking list needs to be modified before Picker 2 can be assigned to pick them up. At this

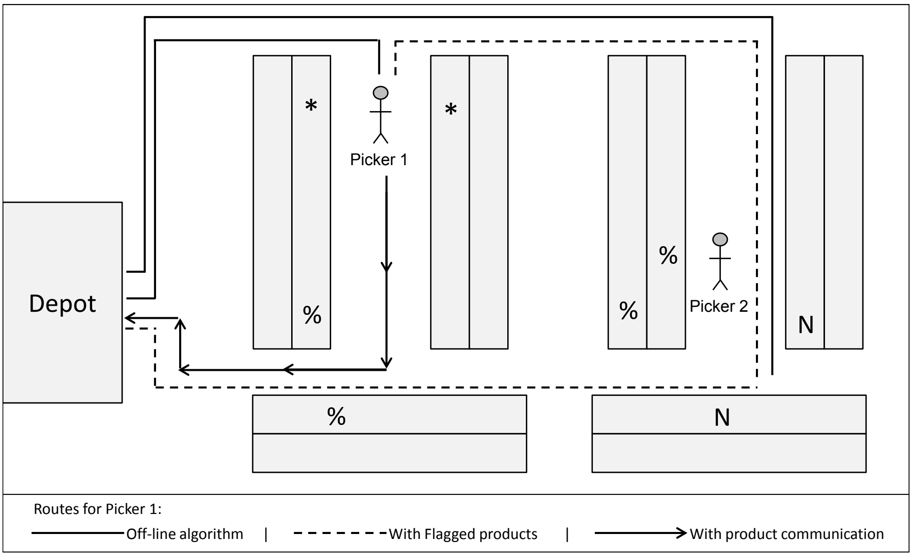


Fig. 4. Dynamic order-picking rescheduling example

point the products will communicate with each other and decide to re-assign the "%" products that are closer to the depot to Picker 1 and the "N" products to Picker 2. As an example, we have drawn the picking routes for Picker 1 only, in the three solutions described above. As mentioned earlier, the off-line solution requires the new order to be picked in a new picking task.

5 Conclusion and Discussion

This paper is a first attempt to identify the key areas in warehouse operations management that the product intelligence paradigm can provide some benefits at. At the same time, it aims to identify the main opportunities and challenges for its adoption in the development of warehouse management systems in real-case scenarios. More specifically, we showed how a WMS using product intelligence can be beneficial for the scheduling and control of the storage location assignment and the picking operations and we argued on the significance of these benefits using a case study of a third-party-logistics warehouse company. Although we have not managed to report quantitative results on the performance of such an approach compared to more conventional, centralised ones, we believe that its impact can be significant firstly, in cases of high uncertainty with a high numbers of unpredictable events affecting the operations and secondly, in cases where there is a need for high customer control over the warehouse's operations.

Another issue that comes out of this study regarding solutions that use the product intelligence paradigm (although not discussed in detail here) is the importance of the communication and/or the negotiation mechanism that the products and the resources participating in the operations will use among them. This element of the system, even if it might seem trivial at first sight, can have a big impact on the performance of a product intelligence approach in real operations as it can affect the final decisions that will be made and executed. Different mechanisms should be developed and tested for each specific case before the adoption of one of them in real industrial systems.

References

1. McFarlane, D., Giannikas, V., Wong, A.C., Harrison, M.: Product intelligence in industrial control: Theory and practice. Annual Reviews in Control 37(1), 69–88 (2013)
2. Meyer, G.G., Främling, K., Holmström, J.: Intelligent products: A survey. Comput. Ind. 60(3), 137–148 (2009)
3. Meyer, G.G., Roest, G., Szirbik, N.: Intelligent products for monitoring and control of road-based logistics. In: International Conference of Management and Service Science, pp. 1–6 (2010)
4. Giannikas, V., McFarlane, D.: Product intelligence in intermodal transportation: The dynamic routing problem. In: 3rd International Conference on Dynamics in Logistics (2012)
5. Shi, K., Chen, X., Qin, X.: Pheromone based self-routing of intelligent products in dynamic manufacturing system. In: Zhang, J. (ed.) ICAIC 2011, Part V. CCIS, vol. 228, pp. 260–268. Springer, Heidelberg (2011)

6. Chiang, D.M.H., Lin, C.P., Chen, M.C.: The adaptive approach for storage assignment by mining data of warehouse management system for distribution centres. Enterprise Information Systems 5(2), 219–234 (2011)
7. Gu, J., Goetschalckx, M., McGinnis, L.F.: Research on warehouse operation: A comprehensive review. Eur. J. Oper. Res. 177(1), 1–21 (2007)
8. Faber, N., de Koster, R., van de Velde, S.L.: Linking warehouse complexity to warehouse planning and control structure: an exploratory study of the use of warehouse management information systems. Int. J. Phys. Distrib. Logist. Man. 32(5), 381–395 (2002)
9. Poon, T., Choy, K., Chow, H.K., Lau, H.C., Chan, F.T., Ho, K.: A rfid case-based logistics resource management system for managing order-picking operations in warehouses. Expert Syst. Appl. 36(4), 8277–8301 (2009)
10. Liu, T., Liu, J., Liu, B.: Design of intelligent warehouse measure and control system based on zigbee WSN. In: International Conference on Mechatronics and Automation, pp. 888–893 (2010)
11. Chow, H.K., Choy, K.L., Lee, W., Lau, K.: Design of a RFID case-based resource management system for warehouse operations. Expert Syst. Appl. 30(4), 561–576 (2006)
12. Gu, J., Goetschalckx, M., McGinnis, L.F.: Research on warehouse design and performance evaluation: A comprehensive review. Eur. J. Oper. Res. 203(3), 539–549 (2010)
13. McFarlane, D., Sarma, S., Chirn, J.L., Wong, C., Ashton, K.: Auto ID systems and intelligent manufacturing control. Eng. Appl. Artif. Intel. 16(4), 365–376 (2003)
14. Kärkkäinen, M., Holmström, J., Främling, K., Artto, K.: Intelligent products–a step towards a more effective project delivery chain. Comput. Ind. 50(2), 141–151 (2003)
15. Kim, B.I., Graves, R.J., Heragu, S.S., Onge, A.S.: Intelligent agent modeling of an industrial warehousing problem. IIE Transactions 34(7), 601–612 (2002)
16. Rubrico, J., Ota, J., Higashi, T., Tamura, H.: Scheduling multiple agents for picking products in a warehouse. In: IEEE International Conference on Robotics and Automation, pp. 1438–1443 (2006)
17. Rubrico, J.I., Higashi, T., Tamura, H., Ota, J.: Online rescheduling of multiple picking agents for warehouse management. Robot. Cim.-Int. Manuf. 27(1), 62–71 (2011)
18. Lu, W., Giannikas, V., McFarlane, D., Hyde, J.: The role of distributed intelligence in warehouse management systems. In: 3rd International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing and Robotics (2013)
19. Koster, R.D., Le-Duc, T., Roodbergen, K.J.: Design and control of warehouse order picking: A literature review. Eur. J. Oper. Res. 182(2), 481–501 (2007)

BarterCell: An Agent-Based Bartering Service for Users of Pocket Computing Devices

Sameh Abdalla, David Swords, Anara Sandygulova, Gregory M.P. O'Hare, and Paolo Giorgini

University College Dublin (UCD)
Belfield, Dublin 4, Ireland
And, University of Trento, Italy
{anara.sandygulova,david.swords}@ucdconnect.ie,
{sameh,gregory.ohare}@ucd.ie,
paolo.giorgini@unitn.it

Abstract. The rising integration of pocket computing devices in our daily life duties has taken the attention of researchers from different scientific backgrounds. Today's amount of software applications bringing together advanced mobile services and literature of Artificial Intelligence (AI) is quite remarkable and worth investigating. In our research, software agents of BarterCell can operate in wireless networks on behalf of nomadic users, cooperate to resolve complex tasks and negotiate to reach mutually beneficial bartering agreements. In this paper, we introduce BarterCell that is an agent-based service application for users of pocket computing devices. We introduce new negotiation algorithms dedicated to bartering services in specific. We examine our approach in a scenario wherein it is essential for a multi-agent system to establish a chain of mutually attracted agents seeking to fulfill different bartering desires. And, we demonstrate and analyze the obtained results.

1 Introduction

Pocket Computing Devices (PCDs) such as Smartphones are increasingly showing the efficiency of relying on them and the importance of having them. Now, people are using different types of lightweight PCDs that allow them to check their emails, exchange faxes, surf the Internet, edit documents, do shopping, and play a role in a social network. Agents' deployments in industrial and profit-making applications are continually growing and, related research are relatively expanding (for an overview; [11,10,8]). Accordingly, the literature of Multi-Agent Systems (MAS) as well is witnessing the success of delivering advanced mobile services to users of PCDs, (e.g., Kore [3], mySAM [4], Andiamo [1]).

In an intersection between Distributed Problem Solving (DPS) [6,5] and Multi-agent Systems (MAS) [14,16], our main focus comes in a place related to the efficiency of the negotiation approaches provided to a set of interacting software agents. Several negotiation models were proposed by scholars to introduce proper negotiation protocols, mechanisms, strategies, or tactics that agents may

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 236–245, 2013.

employ to reach mutually beneficial agreement. An example of that can be the *strategic negotiation in multiagent environments* presented in [9], and also those presented in [17,12,13]

Bartering is a disappearing type of trade where items of similar value are exchanged. "*Swapping*" is the modern approach to bartering, which is seen these days by means of websites that encourage end-users to build virtual communities and share similar interests. BarterCell is our approach to provide users of lightweight PCDs a bartering service on the go. Based on the location and characteristics of a specific community, BarterCell would use agents to build the chain-of-exchange that connects several interested frequenters of the same area.

This paper is organized as follows. Section 2 introduces the general architecture of BaretCell. Section 3 introduces the negotiation algorithms we propose for building the chain of mutually beneficial barters. In section 4 we describe the testing environment we evaluated BarterCell within. In section 5 we show the initial results we obtained.

2 BarterCell: Architecture

The architecture of BarterCell, as shown in figure 1, relies on users' capable devices or PCs to accomplish a successful bartering task. Via the pre-installed Java client-application, users create their own profiles using an interface allowing them to insert their service preferences, and add details related to the kind of items they are exchanging. Then users are asked to directly upload the saved data to a central agents platform that, in return, make it available to other agents. Different from the carpooling service application we presented in [1] wherein Jade [2] was used, for the central platform in BarterCell is running Jack [15], which is an interactive platform for creating and executing multi-agent systems using a component-based approach.

Our architecture relies on distributed Bluetooth access points located within a specific environment, (i.e., university), to receive inputs. Therefore, because of technology limitations, users are asked to be present within the coverage of a connecting spot to transmit their data to the central server. Once received from a user, the message or file content is made available to the multi-agent system, thus it can create a delegated agent that carries the particular characteristics of an end-user. This agent is identified using the Media Access Control (MAC) address of the device used to communicate its user's data. On behalf of users, agents start to interact, cooperate and negotiation with each other in order to achieve the predefined objectives in the given time frame.

Among other benefits, JACK was chosen to handle all of agents' interactions because of its ability to meet the requirements of large dynamic environments, which allow programmers of agents to enrich their implementations with the possibility to compatibly access several of the system resources. JACK has also made the communication language applied among involved agents with no restrictions, which made any high-level communication protocol such as KQML [7] or FIPA ACL easily accepted by the running architecture.

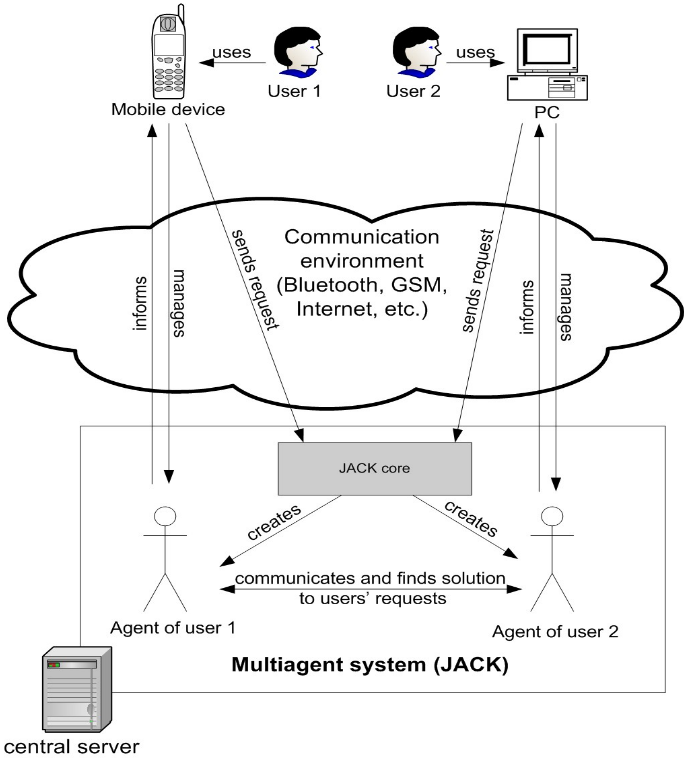


Fig. 1. The architecture of BarterCell

3 BarterCell: Algorithms

The algorithms we present here are considering offered items names, price (estimated by owner), short textual description, time to start offer, period of time to offer and type of the item. Therefore, we assume that users who are using BarterCell own a capable mobile device with a client application installed in it. This end-user client will serve as an interface for a user to access the server-side application where our algorithm has been implemented. Then, upon a user's request, a personal agent reflecting this request is then created. This agent will then be registered within the central multi-agent system and stay active until different action is passed.

The agent created uses these variables: list of demands for all agents of a given system (`cDList`), list of offers for all agents of a given system (`cOList`), ID of an agent that will make bartering chains (`ChainMaker`), most demanded item in a system at a given time (`currentMDItem`), ID of an agent running (`currentAgent`), list of all available agents (`agentsList`), set of agents offering most demanded item (`dG`), the set of agents seeking for most offered item (`oS`), set of agents that are able to make an optimal bartering chain in a given system at particular time (`optimalChain`).

Algorithm 1. BarterCell ChainMaker Selection Algorithm

```
1: currentAgent = agentID
2: while currentAgent ¿ 0 do
3:   cDList = cOList = currentMDItem = MOItem = ChainMaker = NIL
4:   currentChainDecision = optimalChain = NIL
5:   agentsList = getAvailableAgents()
6:   for all a_i ∈ {agentList - currentAgent} do
7:     send(a_i, currentAgent.offers, currentAgent.demands)
8:     cDList = updateCommonDemandsList(a_i.demands, cDList)
9:     cOList = updateCommonOffersList(a_i.offers, cOList)
10:  end for
11:  currentMDItem = findMostDemandedItem(cDList)
12:  dG = findMDGivers(currentMDItem, cOList)
13:  MOItem = findMostOfferedItem(cOList)
14:  oS = findMOSeekers(MOItem, cDList)
15:  ChainMaker = ChainMaker(cDList,cOList,currentMDItem,dG,oS)
16:  if ChainMaker == currentAgent then
17:    runChainMakerService(agentsList,cDList,cOList,currentMDItem,dG,oS)
18:  else
19:    T = initTimer()
20:    while agentProvidesService(ChainMaker,T) do
21:      optimalChain = getResults(ChainMaker,T)
22:      if currentAgent ∈ optimalChain then
23:        userCurrentDecision = sendResults(optimalChain,currentAgent.user)
24:        currentChainDecision=getCommonDecision(optimalChain,agentsList,currentAgent)
25:      else
26:        if currentChainDecision ==”Yes” then
27:          updateAgentODLists(optimalChain,currentAgent.offers,currentAgent.demands)
28:          sendChainContactsToUser(optimalChain,agentsList,currentAgent.user)
29:        end if
30:        sendOptChainDecision(ChainMaker,currentChainDecision)
31:      end if
32:    end while
33:  end if
34: end while
```

As seen in algorithm 1, an agent first tries to get to know other agents available in the system (Line 5). If all other agents will be already involved in a current process of chain creation, the newly arrived agent will seek the ID of the system's `ChainMaker` and notify it of its availability for bartering. The `ChainMaker` will finalize its ongoing computational cycle and inform all agents of service finish (introduced in algorithm 2), then all agents will start a new search for optimal bartering chain.

The new cycle of bartering chain creation will start again from attempting to discover all available agents in the system (Line 5) and creation of list of these agents. For each available agent, a step is taken to communicate its demanded and offered items; (Lines 6-10). Thus all agents of the system will have common demands list (`cDList`) and common offers list (`cOList`). Based on the list of common demands, each agent finds the most demanded item in a given group of agents at given time (`currentMDItem`) and the corresponding set of agents that proposes that item; (Line 12). The agent with the most offered item other agents are seeking is then selected to further define `ChainMaker`; (Line 15).

Here, if an agent is selected to be the `ChainMaker`; (Line 16), then a simple algorithm run showing this agent's acceptance of the role of `ChainMaker` and thus providing other agents with corresponding service. Alternatively, if other agent was chosen for this role then other agents are informed to be tracking responses from `ChainMaker`; (Lines 19-31). However, tracking ChainMaker's responses, following results, and also checking for service availability are made in (Line 20). This function is designed for both parsing of messages from `ChainMaker` and checking whether it can carry out its role. Every time a `ChainMaker` finishes creating an optimal chain, it notifies all agents involved and those agents that are out of it.

Timer initialization, (Line 19), is made to check ChainMaker's availability to agents that are not interacting for a predefined time. If an agent is notified by the ChainMaker that it belongs to an optimal chain, (Line 22), it will then reports to the user it delegates whether the current status is acceptable or no by giving the chain; (Line 23). Eventually, newly selected ChainMaker starts building its list of rejected bartering chains. Having a positive decision as for proposed optimal chain, agents will send to their users contacts of other users whom they should contact in order to make barter (Line 27).

In algorithm 2, the `ChainMaker` start giving its service if it has non-empty list of own demands; (Line 3). Provided that it has the list, ChainMaker starts new computational cycle; (Lines 3-56). The cycle starts with a search for new agents; (Line 5), that might wait to join existing group of agents, which are in agentsList. If there will be at least one agent waiting to join, the ChainMaker will inform all known agents of service finish (Lines 7-9). All agents, including new, will start negotiation process from the beginning (`BarteringService Builder`).

If there are no new agents, ChainMaker will inform all agents of a new computational cycle, and then checks for optimal chains in `queuedChains[]`; (Line 16), that it has proposed during previous computational cycles (if there were any). If at least one user has previously refused to barter in a queued chain, this chain will then be considered as refused and it will never be proposed again by the current ChainMaker. Refused chains will be stored in a `refusedChains[]` set that will be updated along with `queuedChains[]` every time the ChainMaker gets information of refused chain; (Lines 18-22). Accepted bartering chains will simply be removed from `queuedChains[]`; (Line 23).

Assuming that the ChainMaker will now have a list of available agents and a TreeRootAgent, it will then start building bartering trees. Each tree will begin from treeRootAgent with every child, representing agent that demands at least one item from list of its parent's offers. While analyzing every path on such tree the ChainMaker will find repetitions of agents, it will create a complete set of agents that can barter between them. The shortest possible chain will be recorded to `chains[]`, which will consist of shortest bartering chains of three types (combinations of demand types): 1) Strict; 2) Strict + Flexible; 3) Strict + Flexible + Potential. Eventually, the shortest chain selected is built for each corresponding combination; (Lines 25-27).

If a ChainMaker will succeed to find more than one short chain, it will select the optimal one from `chains[]`; (Line 28). Considering chains of equal length, the highest selection priority is given to a chain that will be based on Strict demands while the least priority is given to a chain that will be based on Strict + Flexible + Potential demands. Assuming that an optimal bartering chain is found, then the ChainMaker will inform all concerned agents of being fulfilled; (Line 30). Each agent in the chain will have information such as *which other agents are involved into proposed optimal chain* and *which items should be exchanged and corresponding contact information of users*. The ChainMaker will then remove, from the *common demands list* and *common offers list*, those items that are already chosen to be in proposed in the optimal chain (and will be potentially exchanged later); (Lines 31-32).

If one of optimal chains will be refused to be executed, ChainMaker will restore items that were involved into it ;(Lines 20-21). Every proposed optimal chain will be placed into `queuedChains[]`; (Line 36) to further track whether it will be accepted by users or not. Every agent that will wait for results from ChainMaker and will not be involved into optimal chain, will get a message "cycle finished"; (Lines 37-39). This will be indicator that ChainMaker has finished computing optimal chain, during previous computational cycle that agent was not into it and new computational cycle will be started by the same ChainMaker. This message will cause every agent's timer restart to check chain making service availability.

If ChainMaker fails to achieve a goal, it will notify all involved agents; (Lines 41-43). This message will cause the restart of negotiation process "`BarteringService Builder`". If optimal chain will consist not only of Strict demands items then the ChainMaker tries to make it so by changing treeRootAgent to the next most appropriate agent; (Lines 47-48). In the rest of the algorithm, if there will not be any agent for *current most demanded item*, the next most demanded item and corresponding treeRootAgent will be chosen. If finished with the list of demands or a suspension message received from its user, the ChainMaker will inform all agents of service termination. Agents still interested in bartering service will restart a new negotiation process.

4 BarterCell: Testing Environment

To test our architecture we used a D-Link DBT-900AP Bluetooth Access Point that is connected to the university LAN through a standard 10/100 Mbit Ethernet interface. This device offers a maximum of 20 meters connectivity range with the maximal bit rate support of 723Kbps, and the possibility to concurrently connect up to seven Bluetooth-enabled devices. The same access point is authenticating pocket devices that have BarterCell previously installed in it and, it works as a deliverer of the service requests and responses from and to the

Algorithm 2. BarterCell's ChainMaker Operationing Algorithm

```
refusedChains[] = queuedChains[] = newAgentsQueue = optimalChain = NIL
treeRootAgent = currentAgent
while currentAgent.demands ¡¿ NIL do
   chains[] = NIL
   newAgentsQueue = searchNewAgents(agentsList)
   if newAgentsQueue ¡¿ NIL then
      for all a_i ∈{agentsList - currentAgent} do
         inform (a_i, "service finished")
      end for
      return NIL
   else
      for all a_i ∈{agentsList - currentAgent} do
         inform (a_i, "new cycle start")
      end for
   end if
   for all chain_i ∈ queuedChains[] do
      if hasDecision(chain_i) then
         if chain_i.decision == "No" then
            refusedChains[] = refusedChains[] + chain_i
            cDList = restoreCommonDemandsList(chain_i)
            cOList = restoreCommonOffersList(chain_i)
         end if
         queuedChains[] = queuedChains[] - chain_i
      end if
   end for
   chains[] = findShortestChain(agentsList, treeRootAgent, refusedChains[], "S")
   if chains[] ¡¿ NIL then
      optimalChain = chooseOptimalChain(chains[])
      for all a_i ∈ optimalChain do
         inform (a_i, optimalChain)
         cDList = removeFromCommonDemandsList(a_i.demands, cDList)
         cOList = removeFromCommonOffersList(a_i.offers, cOList)
      end for
      queuedChains[] = queuedChains[] + optimalChain
      for all a_i ∈{agentsList - optimalChain - currentAgent} do
         inform (a_i, "cycle finished")
         cDList = removeFromCommonDemandsList(a_i.demands, cDList)
         cOList = removeFromCommonOffersList(a_i.offers, cOList)
      end for
   else
      for all a_i ∈{agentsList - currentAgent} do
         inform (a_i, "no bartering chain")
      end for
      return NIL
   end if
   if includesFOrPDemands(optimalChain) then
      if existNextAgent(treeRootAgent, currentMDItem, cDList) then
         treeRootAgent = nextAgent(treeRootAgent, currentMDItem, cDList)
      else
         if existNextItem(currentMDItem, cDList) then
            currentMDItem = nextItem(currentMDItem, cDList)
         end if
      end if
   end if
   treeRootAgent = ChainMaker(cDList, cOList, currentMDItem, dG, oS)
end while
for all a_i ∈ {agentsList - currentAgent} do
   inform (a_i, "service finished")
end for
```

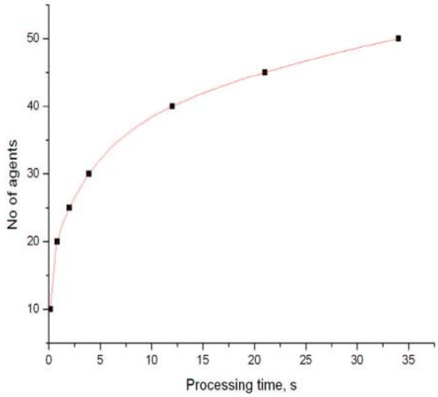


Fig. 2. Simulating the number of Agents

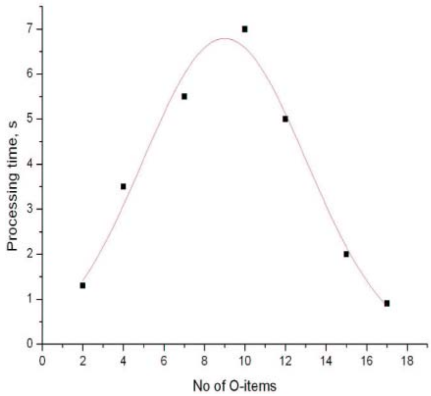


Fig. 3. System Load Distribution

central servers. On the end-user side, four competent cell phones were used to communicate semi-adjusted bartering interests with central servers. These devices are Nokia 6600, 6260, 6630 and XDA Mini. On the server side, a capable PC was used with JACK 5.0 and BlueCove installed in it.

5 BarterCell: Results

While simulation BarterCell we used JDots for tree building, which is object oriented software component. Each node of a tree was built with JDots representing an object with its own fields and methods. Our algorithm works slower with a large number of agents (e.g., ¿300) because the main agent needs to build three trees. Nevertheless, while testing with less than 200 agents, our algorithms are giving efficient results in time.

Figure 2, shows how fast the main agent finishes searching for possible optimal chains depending on the total number of known agents. Here, we assumed that the number of O-items is 5 and the number of D-items is 15.

Figure 3 shows how fast the main agent finish searching for all possible optimal chains depending on number of items that each agent proposes. Total number of offered + desired items is constant (20). Peak of the graph represents the most time consuming state when number of offered items is equal to number of desired items. In this state the main agent has the biggest number of possible exchange combinations. Assumptions used, Max number of items: 20, Max number of D-items: 15, and Number of agents: 30.

Figure 4 shows how fast main agent will finish searching for all possible optimal chains depending on number of items at each known agent. In this run, results here are particular since we assumed that the number of demanded items `D-items` = number of offered items `O-items`, and the number of D-items are only of "Strict" type and the number of agents is equal to 30.

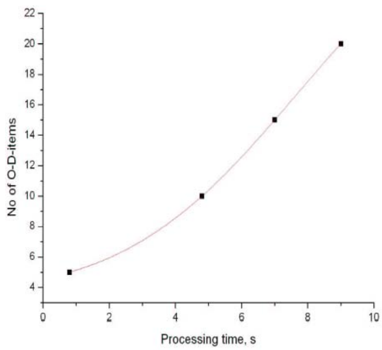


Fig. 4. Simulating the number of items at each agent level

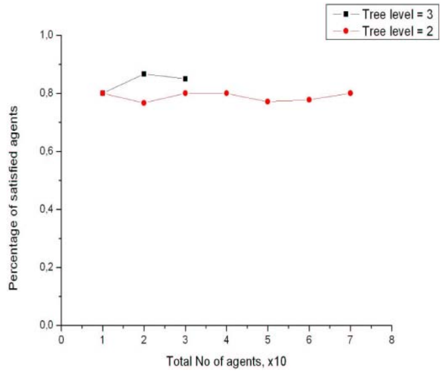


Fig. 5. Agent Satisfaction Level

Figure 5 represents how many agents would be satisfied (i.e. involved in one of optimal chains produced by main agent) until the main agent finishes all possible chain-building processes. Depending on the trees level (depth) the percentage of satisfaction will vary. Here, we assumed that the number of D-items is 20 and the number of O-items is 5.

6 Conclusions

In this paper we introduced BarterCell that is a software architecture for providing location-based bartering service for users of pocket computing devices. This application was developed to: 1) revive the idea of bartering within members of a specific community and promoting the benefits of location-based services, 2) to test new voting-*like* negotiation algorithms for agents representing nomadic users and interacting very actively on-the-go, 3) motivate existing users of pocket devices, and attracting new ones to benefit from recent advanced technologies by widening the range of services that can be offered to them on the go.

Acknowledgment. Authors would like to acknowledge the support of the Science Foundation Ireland (SFI) under the grant 07/CE/I1147 as well as the European regional Development Fund (ERDF) for supporting partially the SC project trough the Ireland Wales Program (INTERREG 4A). Authors would like to thank Oleksiy Chayka for the effort made to implement BarterCell.

References

1. Abdel-Naby, S., Fante, S., Giorgini, P.: Auctions negotiation for mobile rideshare service. In: Proceedings of the Second International Conference on Pervasive Computing and Applications (ICPCA 2007). IEEE, Birmingham (2007)
2. Bellifemine, F., Rimassa, G.: Developing multi-agent systems with a fipa-compliant agent framework. Software - Practice & Experience 31(2), 103–128 (2001)

3. Bombara, M., Calì, D., Santoro, C.: KORE: A multi-agent system to assist museum visitors. In: WOA, Villasimius, CA, Italy, pp. 175–178 (September 2003)
4. Bucur, O., Boissier, O., Beaune, P.: A context-based architecture for learning how to make contextualized decisions. In: Proceedings of the First International Workshop on Managing Context Information in Mobile and Pervasive Environments, Ayia Napa, Cyprus (May 2005)
5. Durfee, E.H.: Distributed problem solving and planning. In: Luck, M., Mařík, V., Štěpánková, O., Trappl, R. (eds.) ACAI 2001. LNCS (LNAI), vol. 2086, pp. 118–149. Springer, Heidelberg (2001)
6. Durfee, E.H., Lesser, V.R., Corkill, D.D.: Trends in cooperative distributed problem solving. IEEE Transactions on Knowledge and Data Engineering 1(1), 63–83 (1989)
7. Finin, T., Fritzson, R., McKay, D.: A language and protocol to support intelligent agent interoperability. In: The Proceedings of the CE&CALS Conference. Morgan Kaufmann, Washington (1992)
8. Jennings, N.R., Crabtree, B.: The practical application of intelligent agents and multi-agent technology. Applied Artificial Intelligence 11(5), 3–4 (1997)
9. Kraus, S.: Strategic negotiation in multiagent environments. MIT Press, Cambridge (2001)
10. Mckean, J., Shorter, H., Luck, M., Mcburney, P., Willmott, S.: Technology diffusion: analysing the diffusion of agent technologies. Autonomous Agents and Multi-Agent Systems 17(3), 372–396 (2008)
11. Munroe, S., Miller, T., Belecheanu, R.A., Pěchouček, M., McBurney, P., Luck, M.: Crossing the agent technology chasm: Lessons, experiences and challenges in commercial applications of agents. The Knowledge Engineering Review 21(4), 345–392 (2006)
12. Sierra, C., Faratin, P., Jennings, N.R.: A service-oriented negotiation model between autonomous agents. In: Padget, J.A. (ed.) Human and Artificial Societies 1997. LNCS (LNAI), vol. 1624, pp. 201–219. Springer, Heidelberg (1999)
13. Smith, R.G.: The contract net protocol: High-level communication and control in a distributed problem solver. IEEE Transactions on Computers C-29(12), 1104–1113 (1981)
14. Weiss, G.: Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. The MIT Press, Cambridge (1999)
15. Winikoff, M.: JACK intelligent agents: An industrial strength platform. In: Bordini, R.H., Dastani, M., Dix, J., Seghrouchni, A.E.F. (eds.) Multi-Agent Programming, ch. 7, pp. 175–193. Springer, US (2005)
16. Wooldridge, M.: An Introduction to MultiAgent Systems. John Wiley & Sons, New York (2002)
17. Zlotkin, G., Rosenschein, J.S.: Negotiation and task sharing among autonomous agents in cooperative domains. In: Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pp. 912–917. ACM, San Mateo (1989)

Negotiating Hour-Wise Tariffs in Multi-Agent Electricity Markets

Fernando Lopes[1,⋆], Hugo Algarvio[1], and Helder Coelho[2]

[1] LNEG−National Research Institute, Est. Paço do Lumiar 22, Lisbon, Portugal
{fernando.lopes,hugo.algarvio}@lneg.pt
[2] University of Lisbon, Bloco C6, Piso 3, Campo Grande, Lisbon, Portugal
hcoelho@di.fc.ul.pt

Abstract. Electricity markets (EMs) are a constantly evolving reality, since both market players and market rules are constantly changing. Two major market models have been considered: pools and bilateral transactions. Pool prices tend to change quickly and variations are usually highly unpredictable. In this way, market participants can enter into bilateral contracts to hedge against pool price volatility. This article addresses the issues associated with the negotiation of forward bilateral contracts. It presents the key features of a negotiation model for software agents and describes a case study involving a 24-rate tariff.

1 Introduction

The electrical power industry was traditionally heavily regulated with a lack of market-price mechanisms. Owing to new regulations, it has evolved into a distributed and competitive industry in which market forces drive electricity prices. Electricity markets (EMs) are not only a new reality but also an evolving one, since both market players and market rules are constantly changing (e. g., the emergence of aggregators).

Two key objectives of EMs are ensuring a secure and efficient operation and decreasing the cost of electricity utilization. To achieve these goals, two major market models have been considered [14]: pools and bilateral contracts. A pool is a market place where electricity-generating companies submit production bids and corresponding market-prices, and consumer companies submit consumption bids. A market operator uses a market-clearing tool, typically a standard uniform auction, to set market prices. Bilateral contracts are negotiable agreements between two parties to exchange electric power under a set of specified conditions, such as price, volume, time of delivery, and duration. Market participants set the terms and conditions of agreements independent of the market operator. They often enter into bilateral contracts to hedge against pool price volatility.

⋆ This work was performed under the project MAN-REM: Multi-agent Negotiation and Risk Management in Electricity Markets (FCOMP-01-0124-FEDER-020397), and supported by both FEDER and National funds through the program "COMPETE−Programa Operacional Temático Factores de Competividade".

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 246–256, 2013.

Practically speaking, opening up electrical energy production to competition is an important tool to improve the efficiency of the electricity industry and therefore to benefit energy customers. Competitive forces can drive companies to innovate and operate in more efficient and economic ways. Innovation can lead to lower prices and better uses of energy resources. However, the analysis of important EMs yields the main observation that they are still far from liberalized. Today there is still a lack of both theoretical and practical understanding and important challenges are still waiting to be addressed more thoroughly. Chief among these are the additional complexities to coordinate technical and economic issues, and the technical difficulties to understand EMs internal dynamics. Stated simply, tariffs do not reflect the pressure of competition.

Multi-agent systems (MAS) are essentially loosely coupled networks of software agents that interact to solve problems that are beyond the individual capabilities of each agent. MAS can deal with complex dynamic interactions and support both Artificial Intelligence (AI) techniques and numerical algorithms. A multi-agent approach in which software agents are capable of flexible autonomous action in order to meet their design objectives is an ideal fit to the naturally distributed domain of a deregulated electricity market. Accordingly, an ongoing study is looking at using software agents with negotiation competence to help manage the complexity of electricity markets towards ensuring long-term capacity sustainability. Specifically, the overall goal of this work is to develop an EM simulator enabling market participants to:

1. negotiate the terms of bilateral contracts, reach (near) Pareto-optimal agreements, and unilaterally de-commit from contracts by paying decommitment penalties;
2. consider dynamic pricing tariffs by pursuing strategies for promoting demand response;
3. manage a portfolio of customers, taking into account trade-offs between the risk and return of bilateral contracts;
4. ally into beneficial coalitions to achieve more powerful negotiation positions, and thus negotiate better tariffs.

This paper is devoted to forward bilateral contracts in electricity markets. It presents the key features of a negotiation model for software agents and describes a case study on forward bilateral contracts—a retailer agent (a seller) and an industrial customer agent (a buyer) negotiate a 24-rate tariff.

This paper builds on our previous work in the area of bilateral contracting in multi-agent electricity markets [4–7]. In particular, it extends the case study presented in [8, 9], by considering a 24-rate tariff and a different set of negotiation strategies. The remainder of the paper is structured as follows. Section 2 presents bilateral contracting in deregulated electricity markets. Section 3 presents the key features of a negotiation model for agents. Section 4 presents a case study on forward bilateral contracts. Finally, concluding remarks are presented in section 5.

2 Bilateral Contracting in Electricity Markets

A bilateral transaction involves only two parties: a buyer and a seller. Depending on the amount of time available and the quantities to be traded, buyers and sellers will resort to the following forms of bilateral trading [2]:

1. *Customized long-term contracts.* These contracts usually involve the sale of large amounts of power (hundreds or thousands of MW) over long periods of time (several months to years). The terms and conditions of the negotiated agreements are set independent of the market operator—although this operator should verify that sufficient transmission capacity exists to complete the transactions and maintain transmission security. The key advantage of these contracts is flexibility, since their terms and conditions are negotiated privately to meet the needs and objectives of both parties. Their disadvantages stem from the cost of negotiation and the risk of the creditworthiness of counterparties.
2. *Trading "over the counter".* These transactions involve smaller amounts of energy to be delivered according to a standard profile, i.e., a standardized definition of how much energy should be delivered during different periods of the day. This form of trading has a much lower cost and is used by producers and consumers to refine their position as delivery time approaches.
3. *Electronic trading.* Market participants can submit offers to buy energy, or bids to sell energy, directly in a computerized marketplace. When a new bid is submitted, the software checks to see if there is a matching offer for the bid's period of delivery. In positive case, a deal is automatically struck and the price and quantity are displayed to all participants. If no match is found, the new bid is added to the list of outstanding bids and remains there until either a matching offer is made, the bid is withdrawn, or it lapses because the market closes for that period. This form of trading often takes place in the minutes and seconds before the closing of the market as generators and retailers fine-tune their position ahead of the delivery period.

3 Two-Party Negotiation

Negotiation, like other forms of social interaction, often proceeds through several distinct phases, notably [3]:

- *a beginning or initiation phase*: focuses on preliminaries to bargaining and is marked by each party's efforts to acknowledge a dispute and to posture for positions;
- *a middle or problem-solving phase*: seeks a solution for a dispute and is characterized by extensive interpersonal interaction toward a mutually acceptable agreement;
- *an ending or resolution phase*: centers on building commitment and implementing a final agreement.

This section presents the key features of a negotiation model for bilateral contracting in electricity markets, focusing on the operational and strategic process of preparing and planning for negotiation and the central process of moving toward agreement.[1] Let $\mathcal{A}=\{a_1, a_2\}$ be the set of autonomous agents (negotiating parties). Both the number of agents and their identity are fixed and known to all the participants. Let $\mathcal{I}=\{x_1, \ldots, x_n\}$ be the negotiating agenda—the set of issues to be deliberated during negotiation. Let $\mathcal{D}=\{D_1, \ldots, D_n\}$ be the set of issue domains. For each issue x_k, the range of acceptable values is represented by the interval $D_k=[min_k, max_k]$.

3.1 Pre-negotiation

Negotiators who carefully prepare and plan will make efforts to perform a number of activities, including:

1. prioritizing the issues;
2. defining the limits and targets;
3. selecting an appropriate protocol;
4. specifying the preferences.

Prioritization involves deciding which issues are most important and which are least important. Target setting involves defining two key points for each issue at stake in negotiation:

1. the *resistance point* or *limit*: the point where every negotiator decides to stop the negotiation rather than to continue, because any settlement beyond this point is not minimally acceptable;
2. the *target point* or *level of aspiration*: the point where every negotiator realistically expects to achieve a settlement.

The negotiation protocol is an alternating offers protocol [10]. Two agents or players bargain over the division of the surplus of $n \geq 2$ distinct issues. The players determine an allocation of the issues by alternately submitting proposals at times in $\mathcal{T} = \{1, 2, \ldots\}$. This means that one proposal is made per time period $t \in \mathcal{T}$, with an agent, say $a_i \in \mathcal{A}$, offering in odd periods $\{1, 3, \ldots\}$, and the other agent $a_j \in \mathcal{A}$ offering in even periods $\{2, 4, \ldots\}$. The agents have the ability to unilaterally opt out of the negotiation when responding to a proposal.

Negotiation starts with a_i submitting a proposal $p^1_{i \to j}$ to a_j in period $t=1$. The agent a_j receives $p^1_{i \to j}$ and can either accept it (`Yes`), reject it and opt out of the negotiation (`Opt`), or reject it and continue bargaining (`No`). In the first two cases the negotiation ends. Specifically, if $p^1_{i \to j}$ is accepted, negotiation ends successfully. Conversely, if $p^1_{i \to j}$ is rejected and a_j decides to opt out, negotiation terminates with no agreement. In the last case, negotiation proceeds to the next time period $t=2$, in which a_j makes a counter-proposal $p^2_{j \to i}$. The tasks just described are then repeated.

[1] This section builds on and updates the material presented in [6, 7, 9].

Definition 1 (Proposal). *Let $\mathcal{A}$ be the set of negotiating agents and $\mathcal{I}$ the set of issues at stake in negotiation. Let $\mathcal{T}$ be the set of time periods. A proposal $p^t_{i\to j}$ submitted by an agent $a_i \in \mathcal{A}$ to an agent $a_j \in \mathcal{A}$ in period $t \in \mathcal{T}$ is a vector of issue values:*

$$p^t_{i\to j} = (v_1, \ldots, v_n)$$

where v_k, $k = 1, \ldots, n$, is a value of an issue $x_k \in \mathcal{I}$.

Definition 2 (Agreement, Possible Agreements). *An agreement is a proposal accepted by all the negotiating agents in $\mathcal{A}$. The set of possible agreements is:*

$$\mathcal{S} = \{(v_1, \ldots, v_n) \in \mathbb{R}^n : v_k \in D_k, \text{ for } k = 1, \ldots, n\}$$

where v_k is a value of an issue $x_k \in \mathcal{I}$.

Negotiators should express their own preferences to rate and compare incoming offers and counter-offers. Let $\mathcal{I} = \{x_1, \ldots, x_n\}$ be the agenda and $\mathcal{D} = \{D_1, \ldots, D_n\}$ the set of issue domains. We consider that each agent $a_i \in \mathcal{A}$ has a continuous utility function: $U_i : \{D_1 \times \ldots \times D_n\} \cup \{\texttt{Opt}, \texttt{Disagreement}\} \to \mathbb{R}$. The outcome `Opt` is interpreted as one of the agents opting out of the negotiation in a given period of time. Perpetual disagreement is denoted by `Disagreement`.

Now, the additive model is probably the most widely used in multi-issue negotiation: the parties assign numerical values to the different levels on each issue and add them to get an entire offer evaluation [13]. This model is simple and intuitive, and therefore well suited to the purposes of this work.

Definition 3 (Multi-Issue Utility Function). *Let $\mathcal{A} = \{a_1, a_2\}$ be the set of negotiating agents and $\mathcal{I} = \{x_1, \ldots, x_n\}$ the negotiating agenda. The utility function U_i of an agent $a_i \in \mathcal{A}$ to rate offers and counter-offers takes the form:*

$$U_i(x_1, \ldots, x_n) = \sum_{k=1}^{n} w_k \, V_k(x_k)$$

where:

(i) *w_k is the weight of a_i for an issue $x_k \in \mathcal{I}$;*
(ii) *$V_k(x_k)$ is the (marginal) utility function of a_i for x_k, i.e., the function that gives the score a_i assigns to a value of an issue x_k.*

Negotiation may end with either agreement or no agreement. The resistance points or limits play a key role in reaching agreement when the parties have the ability to unilaterally opt out of the negotiation—they define the worst agreement for a given party which is still better than opting out. For each agent $a_i \in \mathcal{A}$, we will denote this agreement by $\hat{s}_i \in \mathcal{S}$. Hence, $\hat{s}_i$ will be the least-acceptable agreement for a_i, i.e., the worst (but still acceptable) agreement for a_i. The set of all agreements that are preferred by a_i to opting out will be denoted by S_i.

Definition 4 (Least-acceptable Agreement, Acceptable Agreements). *The least-acceptable agreement for an agent* $a_i \in \mathcal{A}$ *is defined as:* $\hat{s}_i = (lim_1, \ldots, lim_n)$*, where* lim_k*,* $k = 1, \ldots, n$*, is the limit of* a_i *for an issue* $x_k \in \mathcal{I}$*. The set of acceptable agreements for* a_i *is:*

$$S_i = \{s\colon\ s \in \mathcal{S},\ U_i(s) \geq U_i(\hat{s}_i)\}$$

where $U_i(\hat{s}_i)$ *is the utility of* $\hat{s}_i$ *for* a_i*.*

3.2 Actual Negotiation

The negotiation protocol defines the states (e.g., accepting a proposal), the valid actions of the agents in particular states (e.g., which messages can be sent by whom, to whom, at what stage), and the events that cause states to change (e.g., proposal accepted). It marks branching points at which agents have to make decisions according to their strategies. Concession making strategies have attracted much attention in negotiation research and this article is restricted to them.

Concession Making Strategies. Concession making involves reducing negotiators' demands to (partially) accommodate the opponent. This behaviour can take several forms and some representative examples are now presented. Negotiators sometimes start with ambitious demands, well in excess of limits and aspirations, and concede slowly. *High demands and slow concessions* are often motivated by concern about position loss and image loss [11]. Also, there are two main reactions to the other party's demands and concessions [12]: matching and mismatching. *Matching* occurs when negotiators demand more if their opponents demands are larger or concede more rapidly the faster the opponent concedes. *Mismatching* occurs when negotiators demand more if their opponents demands are smaller or concede more rapidly the slower the opponent concedes.

If mismatching is found at the beginning of negotiation and matching in the middle, a reasonable behaviour for convincing the other party to concede is to start with a *high level of demand* and then to *concede regularly*. Such a "reformed sinner" behaviour is often more effective than a behaviour involving a moderate initial demand and a few additional concessions.

Clearly, bargainers generally view the world differently—they are not identical in their interests and preferences. In particular, they frequently have different strengths of preference for the issues at stake—they place greater emphasis on some key issues and make significant efforts to resolve them favorably. Hence, they concede more often on less important or low-priority issues. *Low-priority concession making* involves changes of proposals in which larger concessions are made on low-priority than on high-priority issues [11].

A formal definition of a negotiation strategy that models some the aforementioned forms of concession making follows (see also [7]). For a given time period $t > 1$ of negotiation, the strategy specifies the concession tactics to be used in preparing counter-offers. It also states whether bargaining should continue or terminate.

Definition 5 (Concession Strategy). *Let $\mathcal{A}$ be the set of negotiating agents, $\mathcal{I}$ the negotiating agenda, $\mathcal{T}$ the set of time periods, and $\mathcal{S}$ the set of possible agreements. Let $a_i \in \mathcal{A}$ be a negotiating agent and T_i its set of tactics. Let $a_j \in \mathcal{A}$ be the other negotiating agent and $p^{t-1}_{j\to i}$ the offer that a_j has just proposed to a_i in period $t-1$. A concession strategy $C_i : \mathcal{T} \to \mathcal{S} \cup \{\texttt{Yes}, \texttt{No}, \texttt{Opt}\}$ for a_i is a function with the following general form:*

$$C_i = \begin{cases} \textit{apply } Y_i \textit{ and prepare } p^t_{i\to j} & \\ \textit{if} \triangle U_i \geq 0 \textit{ accept } p^{t-1}_{j\to i} \textit{ else reject}, & \textit{if } a_j\textit{'s turn and } U_i(p^{t-1}_{j\to i}) \geq U_i(\hat{s}_i) \\ \textit{reject } p^{t-1}_{j\to i} \textit{ and quit}, & \textit{if } a_j\textit{'s turn and } U_i(p^{t-1}_{j\to i}) < U_i(\hat{s}_i) \\ \textit{offer compromise } p^t_{i\to j}, & \textit{if } a_i\textit{'s turn} \quad \textit{(time period } t\textit{)} \end{cases}$$

where:

(i) *for each issue $x_k \in \mathcal{I}$, Y_i is a concession tactic (see below);*
(ii) *$p^t_{i\to j}$ is the offer of a_i for period t of negotiation;*
(iii) $\triangle U_i = U_i(p^{t-1}_{j\to i}) - U_i(p^t_{i\to j})$*;*
(iv) *$U_i(\hat{s}_i)$ is the utility of the least-acceptable agreement for a_i, i.e., the worst (but still acceptable) agreement for a_i.*

Concession Tactics. These tactics are functions that model the concessions to be made throughout negotiation. A formal definition of a generic concession tactic follows (see also [7]).

Definition 6 (Concession Tactic). *Let $\mathcal{A} = \{a_1, a_2\}$ be the set of negotiating agents, $\mathcal{I} = \{x_1, \ldots, x_n\}$ the negotiating agenda, and $\mathcal{D} = \{D_1, \ldots, D_n\}$ the set of issue domains. A concession tactic $Y_i : D_k \times [0,1] \to D_k$ of an agent $a_i \in \mathcal{A}$ for an issue $x_k \in \mathcal{I}$ is a function with the following general form:*

$$Y_i(x_k, f_k) = x_k - f_k(x_k - lim_k)$$

where:

(i) *$f_k \in [0,1]$ is the concession factor of a_i for x_k;*
(ii) *lim_k is the limit of a_i for x_k.*

The following three levels of concession magnitude are commonly discussed in the negotiation literature [3]: large, substantial, and small. To this we would add two other levels: null and complete. Accordingly, we consider the following five concession tactics.

1. *stalemate*: models a null concession on an issue x_k at stake;
2. *tough*: models a small concession on x_k;
3. *moderate*: models a substantial concession on x_k;
4. *soft*: models a large concession on x_k;
5. *accommodate*: models a complete concession on x_k.

Now, concession tactics can generate new values for each issue at stake by considering specific criteria. Typical criteria include the time elapsed since the beginning of negotiation, the quantity of resources available, the previous behavior of the opponent, and the total concession made on each issue throughout negotiation (see, e.g., [1, 4]). In this work, we also consider the quantity of energy traded in a given period of the day. Consider a wise tariff involving $m \in [1, 24]$ periods. Let $a_i \in \mathcal{A}$ be a negotiating agent and E the amount of energy that a_i is willing to trade in a specific period. We model the concession factor f of a_i by the following family of exponential functions [7]:

$$f(E) = exp^{-\beta \, \frac{E}{E_T}}$$

where:

(i) $\beta \in \mathbb{R}^+$ is a parameter;
(ii) E_T is the total amount of energy that a_i is willing to trade in a day.

4 Negotiating Hour-Wise Tariffs: A Case Study

David Colburn, CEO of N2K Power—a retailer agent—and Tom Britton, executive at SCO Corporation—a customer agent—negotiate a 24-rate tariff in a multi-agent electricity market.[2] Table 1 shows the initial offers and the price limits for the two agents. Some values were selected by looking up to real trading prices associated with a pool market in an attempt to approximate the case study to the real-world. In particular, market reference prices were obtained by analysing the Iberian Electricity Market.[3] The minimum seller prices—that is, the limits—were then set to these reference prices. Also, some energy quantities were based on consumer load profiles provided by the New York State Electric & Gas.[4]

Negotiation involves an iterative exchange of offers and counter-offers. The negotiation strategies are the following:

- Starting reasonable and conceding moderately (SRCM): negotiators adopt a realistic opening position and make substantial concessions during negotiation;
- Low-priority concession making (LPCM): negotiators yield basically on low-priority issues throughout negotiation;
- Energy dependent concession making (EDCM): negotiators concede strategically throughout negotiation, by considering the amount or quantity of energy traded in each period of the day;
- Conceder [1]: negotiators make large concessions at the beginning of negotiation and low concessions near a deadline; they quickly go to their resistance points;

[2] As stated earlier, this section extends the case study presented in [8, 9].
[3] www.mibel.com
[4] www.nyseg.com

Table 1. Initial offers and price limits for the negotiating parties

	Consumer			Retailler	
Hour	Price (€/MWh)	Limit (€/MWh)	Energy (MWh)	Price (€/MWh)	Limit (€/MWh)
1	42.26	54.69	6.28	57.18	47.23
2	32.85	42.52	6.03	44.45	36.72
3	31.49	40.76	5.90	42.61	35.20
4	31.45	40.70	5.86	42.55	35.15
5	31.15	40.32	6.00	42.15	34.82
6	31.45	40.70	6.30	42.55	35.15
7	38.36	49.64	7.34	51.90	42.87
8	45.51	58.89	8.97	61.57	50.86
9	43.52	56.32	10.30	58.88	48.64
10	45.51	58.89	11.09	61.57	50.86
11	47.44	61.39	11.50	64.18	53.02
12	45.51	58.89	11.79	61.57	50.86
13	45.56	58.96	11.50	61.64	50.92
14	45.51	58.89	11.44	61.57	50.86
15	42.54	55.06	11.21	57.56	47.55
16	41.65	53.90	10.75	56.35	46.55
17	34.31	44.41	9.99	46.43	38.35
18	32.47	42.02	9.29	43.93	36.29
19	31.08	40.23	8.86	42.06	34.74
20	34.00	44.00	8.76	46.00	38.00
21	42.26	54.69	8.65	57.18	47.23
22	45.22	58.52	8.02	61.18	50.54
23	45.31	58.63	7.27	61.30	50.64
24	43.03	55.68	6.75	58.21	48.09

Table 2. Benefit of agents in the final agreement: (Consumer; Retailer) pairs

	SRCM	LPCM	EDCM	Conceder	Boulware	TFT
SRCM	(0.55; 0.45)	(0.54; 0.47)	(0.47; 0.54)	(0.90; 0.10)	(0.39; 0.61)	(0.68; 0.32)
LPCM	(0.75; 0.27)	(0.77; 0.24)	(0.64; 0.38)	(1.00; 0.01)	(0.53; 0.50)	(0.69; 0.33)
EDCM	(0.54; 0.50)	(0.54; 0.50)	(0.47; 0.57)	(0.85; 0.19)	(0.47; 0.57)	(0.62; 0.42)
Conceder	(0.11; 0.89)	(0.09; 0.93)	(0.08; 0.92)	(0.60; 0.40)	(0.03; 0.97)	(0.73; 0.28)
Boulware	(0.63; 0.37)	(0.62; 0.38)	(0.55; 0.41)	(0.93; 0.07)	(0.34; 0.63)	(0.63; 0.37)
TFT	(0.63; 0.37)	(0.57; 0.44)	(0.52; 0.52)	(0.62; 0.40)	(0.59; 0.43)	(0.55; 0.50)

- Boulware [1]: negotiators maintain the offered values until the time is almost exhausted, whereupon they concede up to the reservation values;
- Tit-For-Tat (TFT) [1]: agents reproduce, in percentage terms, the behavior that their opponent performed $\alpha > 1$ received proposals ago;

We have taken up the possible pairs of strategies, one at a time, and examined their impact on the negotiation outcome. The main response measure was the joint benefit provided by the final agreement, i.e., the sum of the two agents benefits in the final agreement. Table 2 shows the results. The following strategies yielded superior outcomes for both agents:

1. Low-priority concession making (LPCM);
2. Energy dependent concession making (EDCM).

It is important to note, however, that these are *initial* (and *partial*) results in a specific bargaining scenario. There is a need to conduct extensive experiments to evaluate the effect of the aforementioned strategies on both the outcome of negotiation and the convergence of the negotiation process.

5 Conclusion

This paper has presented the key features of a model for software agents that handles two-party and multi-issue negotiation. The paper has also described a case study on forward bilateral contracts. Results from a multi-agent retail market have shown that both low-priority concession making and energy dependent concession making strategies lead to superior outcomes.

References

1. Faratin, P., Sierra, C., Jennings, N.: Negotiation Decision Functions for Autonomous Agents. Robotics and Autonomous Systems 24, 59–182 (1998)
2. Kirschen, D., Strbac, G.: Fundamentals of Power System Economics. Wiley, Chichester (2004)
3. Lewicki, R., Barry, B., Saunders, D.: Negotiation. McGraw Hill, New York (2010)
4. Lopes, F., Mamede, N., Novais, A.Q., Coelho, H.: A Negotiation Model for Autonomous Computational Agents: Formal Description and Empirical Evaluation. Journal of Intelligent & Fuzzy Systems 12, 195–212 (2002)
5. Lopes, F., Mamede, N., Novais, A.Q., Coelho, H.: Negotiation Strategies for Autonomous Computational Agents. In: ECAI 2004, pp. 38–42. IOS Press (2004)
6. Lopes, F., Novais, A.Q., Coelho, H.: Bilateral Negotiation in a Multi-Agent Energy Market. In: Huang, D.-S., Jo, K.-H., Lee, H.-H., Kang, H.-J., Bevilacqua, V. (eds.) ICIC 2009. LNCS, vol. 5754, pp. 655–664. Springer, Heidelberg (2009)
7. Lopes, F., Coelho, H.: Concession Strategies for Negotiating Bilateral Contracts in Multi-agent Electricity Markets. In: IATEM 2012 Workshop and DEXA 2012 Event, pp. 321–325. IEEE Computer Society Press (2012)
8. Lopes, F., Rodrigues, T., Sousa, J.: Negotiating Bilateral Contracts in a Multi-agent Electricity Market: A Case Study. In: IATEM 2012 Workshop and DEXA 2012 Event, pp. 326–330. IEEE Computer Society Press (2012)
9. Lopes, F., Algarvio, H., Coelho, H.: Bilateral Contracting in Multi-agent Electricity Markets: Negotiation Strategies and a Case Study. In: International Conference on the European Energy Market (EEM 2013). IEEE Computer Society Press (2013)
10. Osborne, M., Rubinstein, A.: Bargaining and Markets. Academic Press, London (1990)
11. Pruitt, D.: Negotiation Behavior. Academic Press, New York (1981)
12. Pruitt, D., Carnevale, P.: Negotiation in Social Conflict. Open University Press, Philadelphia (1993)
13. Raiffa, H.: The Art and Science of Negotiation. Harvard University Press, Cambridge (1982)
14. Shahidehpour, M., Yamin, H., Li, Z.: Market Operations in Electric Power Systems. John Wiley & Sons, Chichester (2002)

Context-Aware Self-configuration of Flexible Supply Networks

Alexander Smirnov[1], Leonid Sheremetov[2], Christian Sánchez[3], and Nikolay Shilov[1]

[1] St.Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences, St.Petersburg, Russia
[2] Mexican Petroleum Institute, Mexico-city, Mexico
[3] Autonomous Metropolitan University, Mexico-city, Mexico
{smir,nick}@iias.spb.su, sher@imp.mx,
csanchez@correo.cua.uam.mx,

Abstract. In a complex business, network configuring is a very dynamic process. In advanced supply networks, this task is carried out under time constraints and under uncertainties both in suppliers and in the orders. In this paper we show how semantic service-oriented architectures could assist in multilayer self-configuration of on-demand supply chain channels of a Flexible Supply Network (FSN) inspired in Build-to-Order strategy. The approach is based on the idea to characterize all FSN members by their functions or services and to describe them via profiles thus defining their roles. The profiles described by the application ontology are associated with agent-based services that negotiate in order to dynamically configure a network on demand. Semantic service discovery and dynamic composition are used to accomplish this task.

Keywords: Self-configuration, service oriented architecture, flexible supply network.

1 Introduction

Configuration of virtual organizations is defined as matching of competencies against business opportunities and the selection of the best-fit partners within Virtual Organization Breeding Environment (VOBE) [1]. In constantly changing business environments, supply chain members continually adjust their competencies to the changing conditions. In this case, human analysis of competency information is not effective requiring computer-based mechanisms of Competence Management System (CMS) [2]. Moreover, such mechanisms can be considered as an enabling technology for context-aware self-configuring of FSN. Recently, the concept of VOBE was extended to service-oriented VOBE systematically organized around the concept of service-oriented architecture (SOA) [3-5]. Nevertheless, most of existing competence models, such as UECML (Unified enterprise competence modelling language), SWMRD (Semantic Web-based manufacturing resource discovery) and 4-C (Capabilities, Capacities, Costs, and Conspicuities) [2, 6, 7], are not adapted to collaborative organizations lacking the support for service orientation.

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 257–268, 2013.

To address the challenge of automation of competency-based configuring, the proposed approach integrates the theoretical research in the areas of self-organization, competence profiling, context management, and semantic service composition. In this paper we show how semantic SOA could assist in automation of a dynamic configuration of on-demand supply chain channels of Flexible Supply Network (FSN) inspired in Build-to-Order (BTO) strategy. Self-configuring of FSN is considered as a form of multilayer self-organisation.

The key to achieve such automation relies mainly on solutions of four issues: (1) how to represent the competencies of the FSN member in service description, (2) how to make services interoperable both syntactically and semantically, (3) how to automatically discover, based on the syntactic and semantic descriptions, and select the most appropriate services, and (4) how to assemble them to build the composite service. Thus, the major contribution of this paper is two-fold: i) service-oriented framework for FSN supporting company profile semantic description and production network ontology, and ii) a method for FSN competence-enabled configuring based on suppliers discovery, selection and dynamic service composition. This method is based on previously introduced in [8] query expansion and service matching techniques as a general service discovery method, and proposed in [9] knowledge management platform for FSN preliminarily applied to related problems in [10, 11].

The rest of the paper is organized as follows. In the following section, the idea of context-aware multilayer self-organization is introduced and the conceptual framework is proposed. In Section 3, a conceptual model of the agent as an acting unit of the multilayer self-configuration is developed. Section 4 describes the main stages of self-configuration of FSN based on semantic service composition illustrated by example. Section 5 investigates the implementation and application issues. The advantages and limitations of the proposed approach are discussed in conclusions.

2 Theoretical Background

In order for distributed systems to operate efficiently, they have to be provided with self-organisation mechanisms and negotiation protocols. Below the main concepts of multilayer self-configuration are introduced and conceptual framework is described.

2.1 Context-Aware Multilayer Self-organization

The analysis of literature related to organizational behaviour and team management has showed that the most efficient teams are self-organizing teams working in the organizational context. Self-organisation is the mechanism or the process enabling a system to change its organisation without explicit external command during its execution time [12].

The process of self-organisation of a network assumes creating and maintaining a logical network structure on top of a dynamically changing physical network topology. This logical network structure is used as a scalable infrastructure by various functional entities like address management, routing, service registry, media delivery, etc. The autonomous and dynamic structuring of components, context information and resources is the essential work of self-organisation [13]. The network is self-organised in the sense that it autonomically monitors available context in the network, provides the required context and any other necessary network service support to the requested services, and self-adapts when context changes.

Within the context of this paper, self-organisation is considered as a threefold process of (i) cognition (where subjective context-dependent knowledge is produced) achieved through self-contextualisation, (ii) communication (where system-specific objectification or subjectification of knowledge takes place) implemented via usage of intelligent agents, and (iii) synergetical co-operation (where objectified, emergent knowledge is produced) accomplished due to self-management of the agents and their ability to update internal knowledge depending on the situation. The individually acquired context-dependent (subjective) knowledge is put to use efficiently by entering a social coordination and cooperation process. The objective knowledge is stored in structures and enables time-space distanciation of social relationships.

Social self-organisation has been studied, among others, by Hofkirchner [14], Fuchs [15], etc. who pointed out a significant risk for the group to choose a wrong strategy preventing from achieving desired goals. For this purpose, self-organising groups/systems need to have a certain guiding control from an upper layer. The multi-level context aware self-organization is the basis for the approach presented in the paper.

2.2 Semantic SOA Framework for FSN Configuring

Self-organization of FSN will be considered for a BTO supply network, which general scheme is described below. Once a new order is introduced, a company receives a direct request from the customer order-entry system of an enterprise system. Order promising routes this request instantaneously to all sites that could fulfil the order. Frequently there are multiple, hierarchically ordered members with facilities in different geographic regions. Each network member selects its direct suppliers ("first-level suppliers"). A CMS usually enables locating network members based on their profile and production network ontology [9]. This task is supported by network ontology specifying relationships between the products and product assignment to the production units captured by the “produced by” relationship. Based on this information a decision can be made whether a FSN member is suitable for a particular production channel’s configuration. It is supposed that there is no central control unit that could influence upon a choice of FSN members.

Within the proposed approach the FSN members are represented by sets of Agent and Web services (WS and AS) provided by them (Fig. 1). This makes it possible to replace the configuration of the supply network with that of distributed service network. The proposed framework is based on a layered model of service discovery and composition life-cycle. At the upper layer called Development and Publishing, Service Provider/Developer creates, semantically annotates and publishes services describing companies’ profiles. The intelligent agents constituting the network have self-management and self-contextualisation mechanisms described in the next section.

At the lower layer, once the User’s semantic request is generated, discovery and composition mechanisms come into play in order to generate a ranked list of

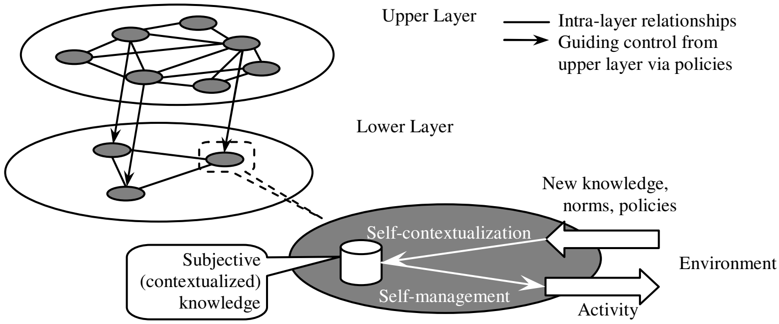


Fig. 1. Generic scheme of the approach

composite services meeting the requirements specified by the request. Once discovery process is over, for the case when a composite service should be generated, service composition takes place resulting in a new on-demand supply chain channel.

Semantic interoperability of service interfaces is grounded in a semantic SOA model. It addresses the problems of matching between requested and provided services, ranking of provided services based on similarity degrees, and selection of the network members with the most similar profiles that satisfy the request.

3 Multilayer Self-configuration Acting Unit: Conceptual Model

Fig. 2 represents the developed conceptual model of the agent as an acting unit of the multilayer self-configuration. The agent has structural knowledge, parametric knowledge, and profile. The agent is characterized by such properties as self-contextualisation, self-management, autonomy, and proactiveness and performs some activities in the community.

Structural knowledge is a conceptual description of the problems to be solved by the agent. This is the agent's internal ontology. It describes the structure of the agent's parametric knowledge. Depending on the situation it can be modified (adapted) by the self-management capability. It also describes the terminology of the agent's context and profile. Parametric knowledge is the knowledge about the actual situation. Its structure is described by the agent's internal ontology, and the parametric content depends on the context of the current situation. It defines the agent's behavior.

Context is any information that can be used to characterize the situation of an entity where an entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves [16]. The context is purposed to represent only relevant information and knowledge from the large amount of those. Relevance of information and knowledge is evaluated on a basis how they are related to a modelling of an ad hoc problem. The context is represented in terms of the agent's internal ontology. It is updated depending on the information from the agent's environment and as a result of its activity in

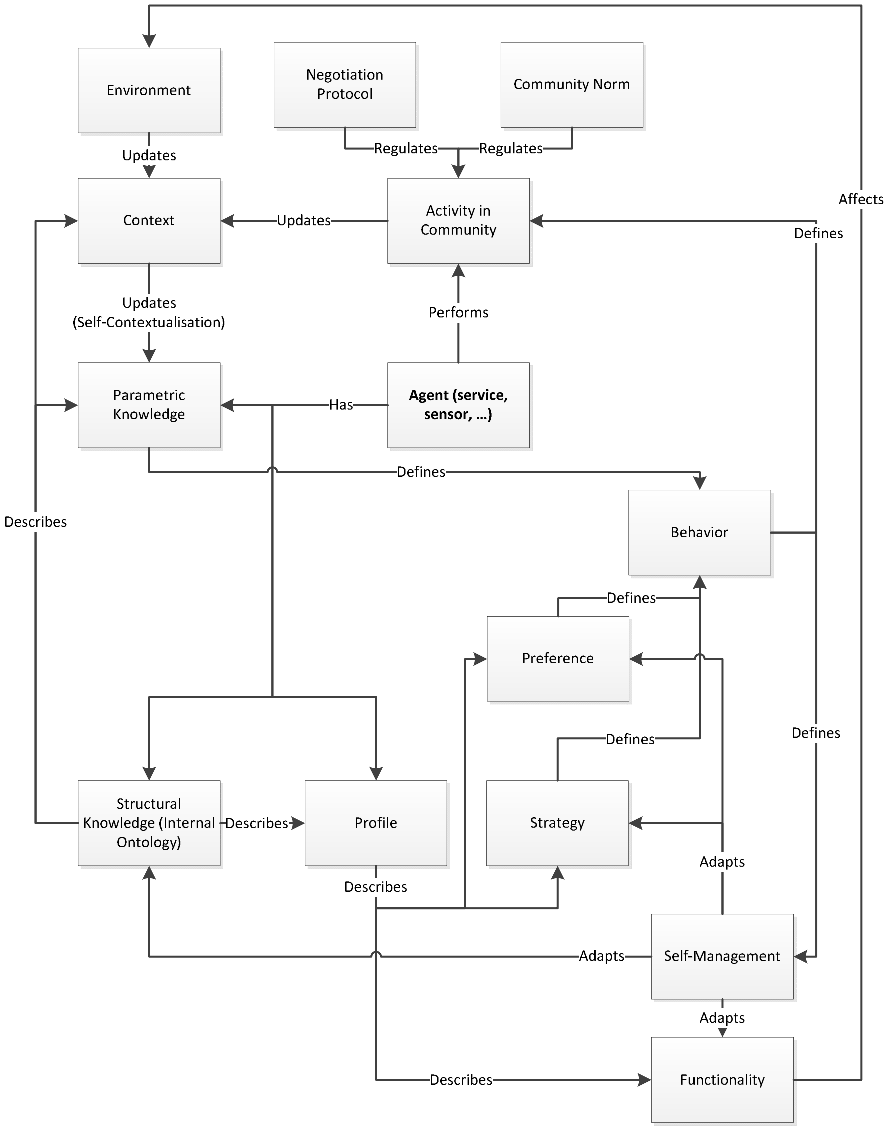


Fig. 2. Multilayer self-configuration acting unit: conceptual model

the community. The context updates the agent's parametric knowledge, which in turn defines the agent's behaviour. The ability of a system (agent) to describe, use and adapt its behavior to its context is referred to as self-contextualization [17]. It is used to autonomously adapt behaviors of multiple agents to the context of the current situation in order to provide their services according to this context and to propose context-based decisions. For this reason the presented conceptual model enables context-awareness and context-adaptability of the agent.

Environment is the surroundings of the system, the agent is a part of, that may interact with the system [18]. The environment affects the agent's context. The agent can affect the environment if it has appropriate functionality (e.g., a manipulator can change the location of a corresponding part).

Functionality is a set of cyber-physical functions the agent can perform. Via the functionality the agent can modify its environment. The agent's functionality can be modified in certain extent via the self-management capability. The functionality is described by the agent's profile.

Profile describes the agent's functionality, preferences and strategies in terms of the agent's internal ontology and in a way understandable by other agents of a cyber-physical system.

Self-Management is an agent's capability achieved through its behavior to modify (reconfigure) its internal ontology, functionality, strategy, and preferences in response to changes in the environment.

Behavior is the agent's capability to perform certain actions (activity in community and/or self-management) in order to change her own state and the state of the environment from the current to the preferred ones. The behavior is defined by the agent's preferences and strategies, as well as by the policies defined on a higher layer of the self-configuration.

Preference is an agent's attitude towards a set of own and/or environmental states and/or against other states. The preferences are described by the agent's profile and affect the agent's behavior. The agent can modify its preferences through self-management.

Strategy is a pre-defined plan of actions rules of action selection to change the agent's own state and the state of the environment from the current to the preferred ones. The strategy is described by the agent's profile defines the agent's behavior. The agent can modify its strategy through self-management.

Activity in community is a capability of the agent to communicate with other agents and negotiate with them through the agent's behavior. It is regulated by the negotiation protocol and community norms.

Negotiation protocol is a set of basic rules so that when agents follow them, the system behaves as it supposed to. It defines the activity in community of the agents.

Community Norm is a law that governs the agent's activity in community. Unlike the negotiation protocol the community norms have certain degree of necessity ("it would be nice to follow a certain norm").

4 Stages of Self-configuration of FSN

This section describes the four stages enabling multilayer configuring of FSN: development and publishing, discovery and selection, service composition and execution (Fig. 3). The service composition scheme is required for the multiagent system to operate since it helps agents finding peers with necessary functions. A simple example from one of the prototype applications illustrates each stage of the configuring.

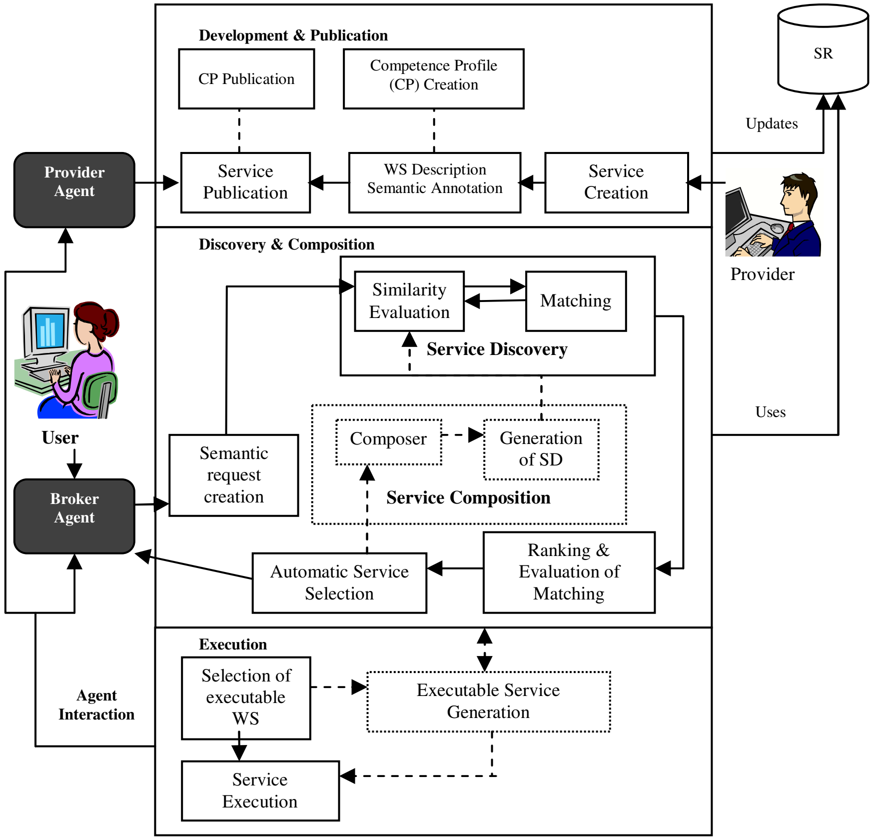


Fig. 3. Three-layered semantic SOA

4.1 Development and Publishing

The first stages start when a provider (agent) wants to generate or enable a new service (increase its functionality) and to make it public. This operation may require also an update of some ontologies (context modification). Once created, the service is annotated semantically (in OWL-S) by the service developer using the available concepts from the existing ontologies (parametric knowledge) to define all its elements (profile, model, grounding). The service is made public updating the service registry (SR) or directory facilitator (DF).

Example. The example from the upstream oil supply chain includes 4 oilfield agents (a1 – a4), a broker agent (ab1) and a Web service w1. The services are registered as shown in Table 1 and Fig. 4. In OWL-S semantic annotation each element describing a service is related with the concepts from available ontologies, e.g. the name of the service is related to ontology of service classification (os1), while the other concepts – to the domain ontologies (do1, do2,….). In the services described in Table 1, upstream oil supply chain ontology is used (parametric knowledge), which can be added at the moment of registry by context modification. Classification ontology includes the concepts related to storage, information and data integration.

Table 1. Registered Services

Agent /Web	Service	Service Description: service_name (Input_Parameters, Output_Parameters)
a1	s1	getProductionData (Well Well_In, Wellbore Wellbore_In, Formation Formation_In, Drilling_Log DrillingLog_In, Water_Analysis WaterAnalysis_In, Treatment Treatment_In, Cleaning Cleaning_In, ProductionData productionData_Out)
a2	s2	getWellLogOperation (Date Date_In, Well Well_In, Wellbore Wellbore_In, Drilling_Log DrillingLog_Out)
a3	s3	getWellInformation (Well Well_In, Wellbore Wellbore_In, Aquifer Aquifer_Out, Water_Analysis WaterAnalysis_Out, Mechanical_Treatment MechTreatment_Out, Cleaning Cleaning_Out)
w1	s4	getCurrentDate (Date Date_Out)

4.2 Discovery and Selection

This stage begins when a user (which one might be an agent), requests some service or resource from the broker agent. The later generates a service request, semantically annotated with OWL-S (using parametric knowledge). When the broker gets the service request he looks for the available services in the SR or DF. The search of the requested service is made considering the different types of service similarity (exact, operationally similar, similar in inputs or in outputs, and different), that help to describe different levels of matching (exact, exact in inputs and outputs, leftover inputs, outputs or inputs/outputs, missing inputs, outputs or inputs/outputs). Similarity types and different matching methods, which can be used for this task, are compared in [8]. Found services are ranked according to their similarity with the request, levels of matching and agent's preferences. The best ranked services are selected in accordance with the agent's strategy. In case the chosen service only partially covers the request, service composition stage is required.

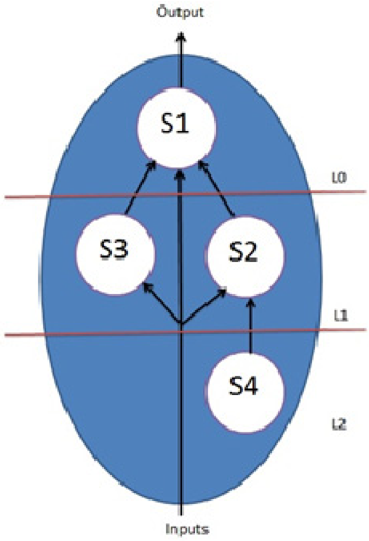


Fig. 4. Composed Service (s5)

Example 1. An oilfield agent (a4) wants to obtain well's production data from operational level, so the request (s5) *"getProductionData (Well Well_In, Wellbore Wellbore_In, ProductionData productionData_Out)"* is sent to the broker agent (ab1). The latter, using the discovery mechanism, finds out that the service (s1) is the most similar (moderately with a coefficient of 1) and therefore she selects it (assuming that she does not have other preferences on the similarity grade). Nevertheless some inputs

(*Formation_In, Treatment_In, …*) are missing to run the service, therefore ab1 will begin the service composition process (to obtain the missing inputs).

4.3 Service Composition

The process of service composition is carried out when the selected service(s) only partially satisfies a request, for example a service lacks inputs to be executed, outputs while being executed, or both. The process of the composition looks for obtaining the lacking elements of the chosen service (inputs and/or outputs) by means of other services. It refers to discovery and selection to form a new service, as described in [11].

The chosen strategy of service composition is the so called backward chaining by layers. First, the first chosen service is selected and its lacking outputs are determined. Once the services providing them are found, the inputs needed by these services are determined and the services that could provide them (input data of the top layer) as outputs (of a low layer). This way the services get connected. The process becomes recursive looking for outputs and lacking input data, until they all are found or one determines that some of them cannot be provided by anyone in the system.

When a compound service is found, the agent can use the acquired knowledge to modify or to re-configure its internal ontology, functionality, strategy and preferences (Self-Management).

Example. Ab1 generates a new service description (to begin the composition and to feed s1), using parametric knowledge, "*GetInformation (Well Well_In, Wellbore Wellbore_In, Formation Formation_Out, Drilling_Log DrillingLog_Out, Water_Analysis WaterAnalysis_Out, Treatment Treatment_Out, Cleaning Cleaning_Out)*". Though s2 and s3 are similar (medium similarity, coefficients 0.92 and 1 respectively), they are lacking of inputs (s2) and outputs (s2 and s3). Ab1 selects s3 and generates a new description to get the missing outputs, "*GetWellInformation (Well Well_In, Wellbore Wellbore_In, Drilling_Log DrillingLog_Out, Production_Log ProductionLog_Out)*", finding s2 like the most similar service but lacking of input data (high similarity, coefficient 0.92).

To find the missing input at this level, ab1 will generate the following description "*GetInformation (Date Date_Out)*" which is very similar to s4 (high similarity, coefficient 1). Since the information is complete, the configuration of the compound service is finished (see Fig. 4). Next time when ab1 is requested for s5, she might change its functionality, strategy and preferences (Self-Management). In the case, a new discovery is not able to find the service that satisfies the request she looks directly for the compound service.

4.4 Service Execution

When the agent is going to run the services its activity in the community begins, firstly it has to identify which of these are WS and which ones are AS, since the latter case implies interaction with other agents and the use of communication protocols, based on the community norm. In case it could not run for some reason any of the required services, the agent begins again the stage of discovery and selection of the services, re-configuring its strategy and preferences (Self-Management).

Example 1. Once ab1 discovered the services by means of which she could satisfy the request, she proceeds to contact the agents that provide them. First the Web service at the highest layer is executed (L2 service s4), followed by agent services of the following levels (L1 and L0, running first s3 and s2 and s1 afterwards) contacted by means of an interaction protocol (fipa-request).

5 Implementation and Application Issues

A practical implementation of the proposed framework is based on interplay of semantic SOA, agents and ontology technologies, which provide a technological framework for FSN. Each supply network member serves as a knowledge source and provides web and/or agent services (WS/AS) for the system to access the owned knowledge. These services are registered at Service Repository (SR) or Directory Facilitator (DF) respectively, the storage places for Web and agent services with an OWL-S description. They are used instead of traditional UDDI to take advantage of discovery, matching and even composition techniques. It makes possible for network member to send a direct request to another network member (without user intervention) through the service portal.

The proposed approach makes use of a broker agent which looks for similar services in different sources. On the way back, this agent receives service descriptions from the service repository for filtering and ranking based on the degrees of semantic similarity. Though the described example considers only one broker agent, each agent associated with FSN member can have broker functionality. Pellet OWL DL reasoner is used for reasoning over OWL and OWL-S ontologies. The use of Pellet for matchmaking may be a bottleneck if all the requests go to the same broker or there are a lot of registered services to be compared with the request. The efficiency of matchmaking is currently under study. Agents are also used as intermediate technology on the one hand, encapsulating discovery mechanisms, and, on the other, merging AS and WS.

The possible application of the proposed approach shows some limitations regarding the flexible suppliers' configuring: the approach can be applied especially to the cases where there is the possibility to attach alternative suppliers quickly to a number of operations within the process. Possible applications of the proposed approach are in the areas of intelligent decision support of FSN and competence management networks. One of such application areas is configuration of Product-Service Systems (PSS). PSS assumes orientation on combination of products and services (often supporting the products) instead of focusing only on products. This is a new paradigm that fits well, for example, industrial equipment manufacturers, for which the equipment maintenance is a considerable part of the business. PSS are flexible by nature: often attaching new services and disconnecting the old ones is required. Hence, the system has to quickly provide available services on the customer request [19].

Another possible application area is Cyber-Physical Systems (CPS), which tightly integrate physical and cyber (IT) systems based on interaction between them in real time. An infomobility support for tourists could be mentioned as a case study [20], which has to integrate various services (transportation, museum and attraction information, weather, etc.) “on-the-fly” in order to provide dynamic multi-modal information to the tourists, both pre-trip and, more importantly, on-trip.

6 Conclusions and Future Work

The paper addresses an important problem of automation of FSN members’ discovery and selection with profile matching by means of semantic integration of services. Discovery techniques permit working with incomplete service descriptions [8]. Company’s profile descriptions in terms of provided services can be of high interest for network members planning cooperation within the FSN and allows automating some of the processes taking place in the business network.

The future development of the presented results is aimed at approaching the MaaS (monitoring as a service) and CPS implementations via development of the appropriate Web-service infrastructure. Implementation of cyber physics systems for business applications (such as supply chain network configuration) requires considering social factors. This can be done via integration with social networks, which is already one of the current trends in the global manufacturing. This, in turn, gives an advantage of using collaboration experience with suppliers that can be used in group recommendation systems in the area of supply chain network configuration. It is also planned to verify the research results in a real life environment.

Acknowledgements. Some parts of the work have been sponsored by grants #12-07-00298-a, #12-07-00302-a, and #11-07-00045-a of the Russian Foundation for Basic Research, project #213 of the research program “Intelligent information technologies, mathematical modelling, system analysis and automation” of the Russian Academy of Sciences (RAS), and project 2.2 “Methodology development for building group information and recommendation systems” of the research program “Intelligent information technologies, system analysis and automation” of the Nanotechnology and Information technology Department of the RAS. The authors appreciate greatly the partial support of this work provided by the IMP, UAM and CONACYT (project grant 153315).

References

1. Camarinha-Matos, L.M., Afsarmanesh, H.: A framework for Virtual Organization creation in a breeding environment. Int. Journal Annual Reviews in Control 31(1), 119–135 (2007)
2. Ermilova, E., Afsarmanesh, H.: Competency modelling targeted on boosting configuration of virtual organisations. Production Planning and Control 21(2), 103–118 (2010)
3. Gou, J., Yang, X., Dai, W.: On Demand Integration of Dynamic Supply Chain Application Based on Semantic Service Oriented Architecture. In: Xu, L., Tjoa, A., Chaudhry, S. (eds.) Research and Practical Issues of Enterprise Information Systems II. IFIP, vol. 254, pp. 589–598. Springer, Boston (2008)

4. Yin, J.W., Zhang, W.Y., Cai, M.: Weaving an agent-based semantic grid for distributed collaborative manufacturing. Int. Journal of Production Research 48(7), 2109–2126 (2010)
5. Paszkiewicz, Z., Picard, W.: Modeling Competences in Service-Oriented Virtual Organization Breeding Environments. In: Shen, et al. (eds.) Proc. 15th Int. Conference on Computer Supported Cooperative Work in Design, pp. 497–502. IEEE (2011)
6. Pepiot, G., Cheikhrouhou, N., Furbringer, J., Glardon, R.: UECML: Unified enterprise competence modelling language. Computers in Industry 58, 130–142 (2007)
7. Cai, M., Zhang, W.Y., Chen, G., Zhang, K., Li, S.T.: SWMRD: a Semantic Web-based manufacturing resource discovery system for cross-enterprise collaboration. Int. J. of Production Research 48(12), 3445–3460 (2010)
8. Sanchez, C., Sheremetov, L.: A Model for Semantic Service Matching with Leftover and Missing Information. In: Xhafa, et al. (eds.) Proc. of the 8th Int. Conf. on Hybrid Intelligent Systems, pp. 198–203. IEEE CS Press (2008)
9. Smirnov, A., Shilov, N., Kashevnik, A.: Developing a knowledge management platform for automotive build-to-order production network. Human Systems Management 27(31), 15–30 (2008)
10. Smirnov, A., Levashova, T., Kashevnik, A., Shilov, N.: Self-Configuration of Flexible Supply Networks: Profile-Based Information Support. In: Proc. of the 13th IFAC Symposium on Information Control Problems in Manufacturing, pp. 97–102 (2009)
11. Sheremetov, L., Sanchez, C.: Semantic SOA 4 SOA Supply Networks. In: Proc. of the 13th IFAC Symposium on Information Control Problems in Manufacturing, pp. 1287–1292 (2009)
12. Di Marzo Serugendo, G., Gleizes, M.-P., Karageorgos, A.: Self-organization in multiagent systems. The Knowledge Engineering Review 20(2), 165–189 (2005)
13. Ambient Networks Phase 2. Integrated Design for Context, Network and Policy Management, Deliverable D10-D1 (2006), `http://www.ambient-networks.org/Files/deliverables/D10-D.1_PU.pdf` (retrieved April 18, 2007)
14. Hofkirchner, W.: Emergence and the Logic of Explanation. An Argument for the Unity of Science. Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering Series 91, 23–30 (1998)
15. Fuchs, C.: Globalization and Self-Organization in the Knowledge-Based Society. TripleC 1(2), 105–169 (2003), `http://triplec.uti.at`
16. Dey, A.K.: Understanding and using context. Personal Ubiquitous Computing 5(1), 4–7 (2001)
17. Raz, D., Juhola, A.T., Serrat-Fernandez, J., Galis, A.: Fast and Efficient Context-Aware Services. John Willey & Sons, Ltd. (2006)
18. Rajkumar, R., Lee, I., Sha, L., Stankovic, J.: Cyber-physical systems: the next computing revolution. In: Proc. of the 47th Annual Design Automation Conference, pp. 731–736 (2010)
19. Shilov, N.: Product-Service System Configuration in SOA-Based Environment. In: Abramowicz, W., Maciaszek, L., Węcel, K. (eds.) BIS 2011 Workshops. LNBIP, vol. 97, pp. 184–195. Springer, Heidelberg (2011)
20. Smirnov, A., Shilov, N., Kashevnik, A.: Ontology-based Mobile Smart Museums Service, Approach for Small & Medium Museums. In: The Fourth International Conference on Advances in Future Internet, pp. 48–54 (2012)

Distributed Resource Search in Self-organising Networks

Muntasir Al-Asfoor[1,2] and Maria Fasli[1]

[1] University of Essex, School of CSEE, Wivenhoe Park, Colchester, UK CO4 3SQ
{mjalas,mfasli}@essex.ac.uk
[2] AL-Qadisiya University, Diwania, Iraq 58002

Abstract. Virtual Organisations (VO) have emerged as an important field of study in the area of Multi Agent Systems (MAS). A VO consists of a number of members that share preferences or goals with the aim of exploiting the available resources and providing better services than a single member would be able to. In this paper, VOs have been used as a way to cluster heterogeneous agents which have semantically close resources together. A VO-based resource search protocol has been developed with search message routing techniques that have been used to forward resource search messages among VOs instead of individual agents. A decision making component has been added to the agent's body to facilitate the process of maintaining the VO′s functionality. Different scenarios have been studied to deal with situations which might affect the VO′s work-flow. The proposed solutions have been implemented and tested in a simulated environment. The simulation results have shown a significant improvement in the search results in terms of the quality of matching, the required time to find the requested resources and the success ratio of requests.

1 Introduction

A Virtual Organisation (VO) consists of a set of autonomous agents each of which may have a range of capabilities and maintains different resources. VOs can provide a useful and effective framework to structure a complex system which is able to solve problems and provide services that an individual agent would not be able to on its own. Coupled with self-organization, which is a dynamic and adaptive process where components of a system acquire and maintain information about their environment and neighbours without external control, VOs can be responsive to the changes in the environment and may remain effective. A fundamental engineering issue when designing self-organizing emergent multi-agent systems (MASs) is to achieve desirable macroscopic properties by manipulating the microscopic behaviour of locally interacting agents [5]. As MAS's get more complex, questions arise about the best way to control agent activity, and thus application performance. Centralised control of MAS is one approach, but is of limited use because of the risk of dependency on the controlling element, and the ensuing lack of robustness. This also makes little sense when agents are autonomous which can be of benefit in various applications. Partially or completely decentralised control is an alternative, but means of implementing this without disrupting agent performance in support of applications is important [1]. The work presented in this paper, considers a network

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 269–280, 2013.

of heterogeneous agents and introduces agent-based organisation formation and management with the local resource description as the main differentiating characteristic for forming organisations. The purpose of such organisations is to improve the resource search in terms of both speed and matching accuracy therefore overall improving the individual agents' performance but also the overall system performance. However, this is a dynamic network, in which agents may connect or disconnect for various reasons or their resources may change. Hence, organisations need to be able to detect and respond to such changes, otherwise the performance of the system and the individual agents will be negatively affected. To this end, we study self-organisation and adaptation solutions that would enable the organisations to respond to change and remain effective.

The rest of the paper is structured as follows. Next we provide, a description of the related topics as well as the work undertaken in the area of self-organised distributed multi-agent systems. Section 3 shows a formal model of the proposed system while section 4 has been devoted to the initial network formation process. Section 5 presents the agent organisation formation algorithm. The proposed scenario for resource sharing among agent organisations and the approach to self-adaptation are explained in section 6. The system simulation, experiments and results are presented in section 7 while section 8 draws the conclusions and discusses avenues for future work.

2 Related Work

With the evolution of the Internet and the need to satisfy new market needs and user demands, new requirements for dynamic resource search have emerged. In an unstructured dynamic environment, self-organisation plays an important role in the process of adapting the network structure with the aim of improving the system performance. Having only partial knowledge about the environment, decentralised search and self-organisation generates new challenges such as dealing with uncertainty or action coordination based only on local states, which cannot be tackled by traditional approaches [2]. Conventional resource search techniques commonly used in decentralised networks where entities are treated equally, do not consider any heuristics about the resources or their locations, instead they use flooding and random walk search algorithms [19]. In order to prevent the resource search message from traversing the whole network and creating excessive network traffic, heuristic resource search has emerged as a promising approach. There has been considerable work by the MAS community on resource search techniques and agent organisations. A dynamic network topology adaptation technique has been proposed by [4] with the aim of improving the search performance of the flooding algorithm by creating semantic communities based on query traffic. In contrast with our work, the authors have focused more on changing the topology based on statistical heuristics by collecting nodes with similar interests together in small communities. They assumed that the file request could be fulfilled by nodes within the same community. Resource search in a multi agent environment has been addressed by [6], who proposed a mechanism in which each agent can maintain a list of k contacts and their available resources. Based on the information which has been stored locally, the agent could direct its search query to a subset of the agent population using traditional search methods (i.e flooding, teeming or random walk). The proposed method formed some sort of network of caches and did not impose any semantics or re-organisation capability in the agents. A decentralised multi-agent system model has been presented in [9] arguing that convergence to an optimal allocation is a consequence of a balance between random selection and preferential selection. Their algorithm has made use of a central registry to ensure efficiency and fairness, however this approach would create a single point of failure and impose in essence centralised control. In contrast to the work described above, the work presented here has made use of the agent organisations to improve the search process. Furthermore, resources and task descriptions have been used along with semantic matching as a way to guide the search messages as well as improve the organisations work-flow. The basic definition of Virtual Organisation (VO) is simple enough: organisations and individuals who bind themselves dynamically to one another in order to share resources within temporary alliances [7]. VOs can be formed and managed automatically by intelligent agents. To support the envisaged automation, agents represent organisations and individuals by providing and requesting resources/services, by wrapping them or by connecting to their published interface [13].

3 Formal Model

In order to explain the proposed resource search and self-organisation algorithms, in this section we provide a formal description of the proposed system. We assume a heterogeneous network, hereafter the *system*, which consists of autonomous individual agents a_i. Each agent has a limited view of the environment and only knows about another agent if and only if they had made an acquaintance before, i.e. a connection. Typically, agent connections imply a cost so each agent would have a limited number of direct connections to other agents. Furthermore, each agent holds resources which can vary in nature from one agent to another and has task(s) to perform that required resources. In its simplest form, resource search is instigated when an agent does not have the necessary ones to perform a task and would need to find suitable resources by using its network connections to other agents.

Definition 1 *(System). The system is a tuple* $< A, C >$, *where* $A = \{a_1, ..., a_n\}$ *is a finite set of agents and,* $C \subseteq A \times A$ *is a set of connections, where each connection* $(a_i, a_j) \in C$ *indicates the existence of a direct relationship between agents* a_i *and* a_j.

The connection between any two agents is bidirectional; the agent has control over its local resources and direct connections. Furthermore, the agent has knowledge about its direct connections (neighbours).

Definition 2 *(Agent). The Data Model (DM) of an individual agent* a_i *is a tuple* $< ID_i, R_i, Ncon_i, Resource_i, S_i, Cdb_i, T_i, Acc_i, Task_i >$

where

- ID_i is a_i's unique identifier which is used to identify the agent when sending it messages or addressing it in any database throughout the network.
- R_i is a list of a_i's roles which starts with "regular" upon creating the agent and builds up while the agent is operating. An agent may have a different role in different groups/structures/organisations.

- $Ncon_i$ is the maximum number of connections agent a_i may have and it is finite.
- $Resource_i$ is agent a_i's local description of resources. In a distributed environment, there have been a range of methods developed to describe resources and tasks. For instance, WSDL (Web Service Description Language) [17], OWL (Web Ontology Language) [16] and RDF (Resource Description Framework) [10] are standard languages that have been developed and can be used for this purpose. For the purposes of this work, and in particular in use in the simulation, and since the focus of this paper is on improving the network traffic performance, resources take the form $Resource = \{RD, Av, Sh, T_s, T_e\}$ where $RD =< r_1, r_2, r_3 >$ is the resource descriptor as the resource consists of three parts which are basically being used to encode a range of resources, but instead of using textual description, they are encoded by numbers each with a value between [0,4]; Av is the available amount; Sh is the the amount currently reserved for use by other agents or tasks by the same agent; T_s is the reservation start time; T_e is the reservation end time;
- S_i represents the resource search technique a_i uses to search for any extra resources when needed. When initiating a search for a needed resource the agent would adopt a specific search technique.
- Cdb_i is a list of tuples of the form <ID, Resource, contact time, connection type > to be used by a_i to store any knowledge it has about its connections, the size of the list is maximum $Ncon_i$.
- T_i is a list of time stamps to keep track of the agent a_i's creation time, activation time, resource update time, task update time, and termination time.
- Acc_i is the required matching accuracy.
- $Task_i$ is the task description which takes the form $TD = \{RD, T_u, R_m, T_s, T_D\}$, where $RD =< r_1, r_2, r_3 >$ is the resource descriptor consisting of three parts each with a value between [0,4]; T_u is the task utility; R_m is the required amount; T_s is the task start time; T_D is the task duration.

4 Initial Network Formation

To test the proposed algorithms, a distributed multi-agent model which represents the dynamic networked system has been designed. The system has been modelled as a population of agents that communicate with each other and with the environment via different types of messages based on what kind of information the agent needs to convey. At the beginning of the simulation, the agent population is created and each agent encapsulates certain internal information which is only known by the agent itself (autonomy). After that, the system starts activating the agents randomly. While the simulation is running more agents will join the system till the saturation condition $M = n$ is reached where M is the number of active agents and n is the number of agents. Upon activating a new agent, the agent starts operating by connecting to the network via sending a $hello$ message to a random active agent. The $hello$ message has the following form:

$hello < To, From, TS, Resource >$ Where, To is the sender's ID; $From$ is the destination's ID; TS is the sending time (simulation cycle) and $Resource$ is the sender's resource description. After connecting to the network, the agent would start executing tasks or sharing its resources. The task creation process and resource description are done by using a random uniform distribution to simulate the fact that the tasks and resources are changing over time. If the agent needs some resources to execute its tasks and it does not own these resources or it has already committed its own resources to another agent and depending on the deadline for executing a task it can send a *R*esource *Search M*essage of the form:

$RSM < To, From, TS, TD, Hops >$

Where: $From$ is the destination's ID; TS is time stamp to record message starting time, TD is the task description, and $Hops$ is the number of hops.

An agent that receives an RSM message and can offer the requested resource, responds with a *R*esource *F*ound *M*essage:

$RFM < To, From, MyResource, TS >$

Where: $MyResource$ is the responding agent's resource description and TS is the time stamp of the message.

5 Organisation Formation

The proposed organisation formation algorithm is centred around the idea of creating a group of agents which are individually autonomous but they have semantically close local resource descriptions within a predefined matching threshold. The assumption is that an individual agent has no global knowledge about other agents in the environment. However, agents can infer information about others while interacting with them which helps an individual agent to build a list of potential contacts and their available resources at the time of interaction. In large expandable networks such as the Internet, a node (agent) cannot have a direct connection to all the agents across the network and besides the number of contacts that an agent can maintain is finite. As resources are being used by agents to carry out their tasks, the agent may need to look further away for much needed resources (away from its local neighbourhood), but if it attempts to do so, then the search may take an inordinate amount of time which means that tasks may not have enough time to be executed. Therefore, if the agents could organise themselves in structures based on this occasion on their resources descriptions, then we hypothesize that the search for suitable resources could be faster and more efficient. An organisation takes the form of $Org < Oid, OH, Members, K >$ where Oid is the unique identifier, OH is the identity of the head of the organisation (by default this is the initiator), $Members$ is a list of the members at a given time, K is the maximum size in terms of members that the organisation can have. The first step in the organisation formation process is for the initiator to invite other agents to join its organisation. To avoid the states in which all the agents join only one organisation which might turn the organisation into a centralised network, we assume that the organisation size is finite. Based on that, the system uses k-graph [8] to assign each organisation a random number K that represents the size. The potential member must fulfil the criteria which are defined by the initiator. As the aim of creating organisations is to improve the resource search process, the main criterion for an agent to be invited to join an organisation is to hold resources which are semantically close to the resources of the initiator within a required threshold (i.e. matching quality). Since the agent has only knowledge about a subset of the network population, then these are the only potential invitees in the first instance. Upon receiving an invitation message from an initiator, the receiver would decide based on its preferences, which might not be the same as the initiator's whether to join. If the decision is to join, then the agent will do the following: first of all create an entry in its local database keeping information about the organisation and secondly send an "I agree to join" message to the initiator. The initiator would then update the organisation's database by creating a new entry with the new member's information. The organisation size could vary based on the required quality of matching with smaller organisations for higher matching quality and vice versa.

Since the resource has been modelled as a tuple $< r_0, r_1, r_2 >$ and each part r_i has a value between [0,4] as explained before, using the Manhattan distance measure this could result to 13 different matching thresholds. Suppose the requester requires resource $< 0, 0, 0 >$ and the available resource is $< 4, 4, 4 >$ then the similarity result is $(|0 - 4| + |0 - 4| + |0 - 4|) = 12$ which represents the case where the resources are completely different (the highest possible error). The other case is when the resources are exactly the same: assume that the requester requires a resource $< 2, 2, 2 >$ and the available resource is $< 2, 2, 2 >$ then the similarity result is $(|2-2|+|2-2|+|2-2|) = 0$ which represents the lowest error (exact matching).

Having agent organisations creates another level of networking which is the connections among organisations. To achieve this level of connection, an organisation needs to be connected to at least one other organisation via their OHs. To establish such a connection, a new OH has to forward a random walk message looking for another OH to connect with.

6 Resource Search Using Agent Organisations

In a resource constrained environment, resource sharing among agents can be deployed for mutual benefit. Agents may share their unused resources with others and in return get a portion of the task's utility. As the network might scale up, searching the whole network becomes practically infeasible in particular as the tasks themselves are time-constrained. To overcome this problem, we have introduced an organisation-based resource search protocol (OBRSP) in which the resource search messages traverse organisations instead of individual agents. If an agent a_i needs some resource, it initiates the OBRSP protocol by sending a resource request to its OH_i. Then OH_i checks if the resource is available locally or within the organisation, if not, OH_i checks its list of organisations and sorts them based on the required resource description using the Manhattan [11] distance measure as follows.

$$OH_j(i) = \underset{j \in LO}{argmax}[Sim(j, i)] \tag{1}$$

Equation 1 calculates the best potential providing organisation by choosing the OH_i with the maximum semantic similarity of resources to the requested resource using Manhattan distance measure. Each OH knows about the average semantic similarity of resources provided by the organisations with which it has a direct connections; it is assumed that the network of OHs is fully connected (i.e. an OH_i is directly connected to at least one other OH_j). As the simulation proceeds, an OH_i will find out about more OHs through resource search messages interactions. Next, OH_i forwards the resource search message to the best matching organisation's OH_j which in turn, chooses the RM_j that holds the closest resource description using the Manhattan measure as well. If the provider agent agrees to share its resource, it contacts the requester directly for further negotiation/agreement. If there are no available resources at all in the selected organisation, then the requester's OH_i contacts the second best matching organisation and so on. However, the search fails in two cases: (i) if the resource search message has expired (has exceeded the maximum time to live TTL predefined by the requester); (ii) if there are no agents left to contact (the RSM has traversed the whole network).

7 Self-organisation Process

Self-Organisation in the context of this work is the process of self-adapting the network structure as well as the organisational membership based on the similarity of resources and the potential utility of connections. As the OH maintains a database of the membership, upon receiving a message from a new agent which it has not had an acquaintance with before, OH checks if the new agent would be beneficial as an organisation member by comparing the potential utility of having it as a member. The utility calculation depends on the semantic similarity between the new agent's resource description (Eq. 2):

$$U(j, i) = sim(R_j, R_i) \tag{2}$$

Where: $U(j, i)$ is the utility of a_j as a potential member from a_i's point of view where a_i is an OH; $sim(R_j, R_i)$ is the semantic similarity between a_j' and a_i's resource descriptions. If the organisation has capacity to accept a new member, a_j will be added to the list of members. However, if $U(j, i) > U(k, i)$ where a_k is the member with the lowest utility and the organisation is full, then OH_i will replace a_k by a_j, otherwise a_j will be added to OH_i's list of acquaintances. An OH could have two lists: (i) the list of the members of the organisation that it is heading and (ii) the list of its own acquaintances.

8 Experiments

We have designed and performed two sets of experiments to study and test (i) the performance of the OBRSP and compare it with a random walk algorithm [6]; (ii) the effects of self-organisation on the overall system performance.

8.1 Simulation Tools

In order to support our investigation into the micro-macro link between the selected search parameters, and their global consequences with respect to our system performance metrics; we have chosen to run multiple simulations of our resource discovery scenario across the available state space as defined by our parameters. A number of agent-based simulation tools exist such as Mason [14], NetLogo [18] and Swarm [12] in addition to specific agent languages and platforms e.g. Jason [3]. From this myriad of tools we have selected to use PreSage [15] a Java based rapid prototyping and simulation tool developed to support the animation of agent societies. Similarly to the platforms and tools mentioned above PreSage allows us to script a series of simulation runs with varying parameters, and record results from a global perspective. We can also: *(i)* conditionally trigger exogenous events such as changes to the network;*(ii)* quickly define and adapt our agent communication protocols such as our resource search in a flexible manner;*(iii)* allow individual agents to control the application layer network (from a localised perspective);

8.2 Experimental Set-Up and Simulation Results

To evaluate the proposed solutions and their performance, several experiments have been performed. The following performance metrics have been used:

- Semantic Matching Accuracy: To study the trade-off between the matching quality and the search performance. For simulation purposes we have run multiple simulations with different matching accuracy in the range of $[30\%..100\%]$.
- Average Number of Hops to Success: The average number of hops a resource search message needs to be forwarded through the network to find the required resource successfully.
- Success rate: The percentage of tasks successfully completed.
- System MAE: The System mean average error is a comparison between the required matching quality and the actual found matching quality hence some search messages might end up finding resources with better quality of matching than the requester's requirement.

The network of agents has been configured with the following parameters:

- Network Size: multiple simulations have been run under network size of 300, 400, 500, 600 agents; each agent at the time of activation has at least one connection.
- Search Message Time TO Live (TTL): configured with the value of 10 hops.
- Distribution of Tasks and Resources: a uniform random Gaussian distribution has been used to distribute tasks and to update resources (to simulate the dynamic system behaviour). The uniform random Gaussian distribution parameters have been set as follows:
  - mean time to task = 10 (i.e. on average the agent is issued with a new task every 10 time cycles);
  - variance time to task = 4 (i.e. the variety of randomisation);
  - mean time to resource change = 50 (i.e. on average the resource description changes every 50 time cycles);
  - variance time to resource change = 20 (i.e. the variety of randomisation).
- Maximum neighbourhood size: maximum number of direct connections an individual agent could hold = 10.

The first set of experiments has been run to analyse the efficiency of the OBRSP and we evaluated different configuration parameters and their effects on the overall system performance. We have compared the OBRSP against another search algorithm from the literature, namely, the Random Walk [6] algorithm. We mainly focused on the effects of different parameters on the percentage of tasks successfully completed and the number of steps to find the resource successfully. In these experiments, we mainly analysed the influence of the matching threshold on the search performance as well as the effect of having agents cluster themselves in organisations. The simulation results have shown that having agent organisations improves the search performance as shown in Figure 1. With high matching quality OBRSP performed better in terms of number of hops to find the required resources successfully. However, when the matching quality decreases, the two protocols start to perform relatively close. The explanation of this lies in the Random Walk algorithm's behaviour where the probability of finding the required resource is high with low matching quality(with low matching threshold the requester would accept any available resource). Furthermore, as the OBRSP needs at least three hops to perform the resource search protocol (this is not the case for random walk), with very low matching quality ($< 40\%$), the random walk performed better as it needed in average less than 3 hops to succeed.

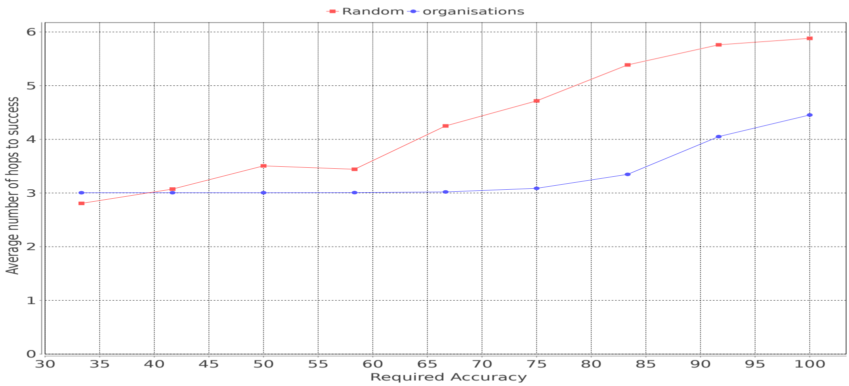


Fig. 1. Number of Hops to Success: Random Walk vs OBRSP

Another experiment has been done to study the percentage of successful tasks under the same configurations for the two protocols (random walk and OBRSP). As the system under consideration is a dynamic expandable network, the main parameters that affect the search success/failure ratio are: search message time to live (TTL) and the required matching quality. With a configuration of $TTL = 10$ and all range of possible matching thresholds, simulation results have shown 0% failure. However, when using the random walk algorithm and when the matching threshold increases, the percentage failure increases and vice versa as shown in Figure 2.

The last set of experiments aimed at studying the ability of organisations to self-adapt in response to environmental changes or to improve the system performance. A comparison study has been done between OBRSP without self-organisation and with

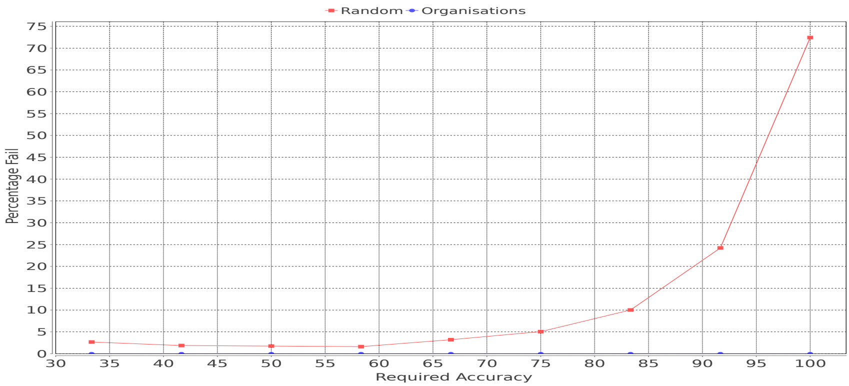


Fig. 2. Percentage Failure: Random Walk vs OBRSP

self-organisation capability. The main parameters for this experiment were the average number of hops to find the resources successfully and the quality of these resources. The quality of resources has been measured as the mean average error (MAE) between the requested resource and the actual provided resource. Simulation results have shown that in terms of number of hops to success, OBRSP with self-organisation capability performs better with high matching quality. However, they both perform relatively the same with low matching quality as both OBRSP and the one with added self-organisation capability need at least 3 hops to perform the search as shown in Figure 3.

Fig. 3. Number of Hops to Success: OBRSP vs OBRSP with Self-Organisation

The OBRSP with self-organisation has shown better performance in terms of MAE in comparison to native OBRSP as shown in Figure 4. With high matching threshold, MAE must be very small (close to 0) so both settings perform the same, however with lower matching threshold OBRSP with self organisation performs better.

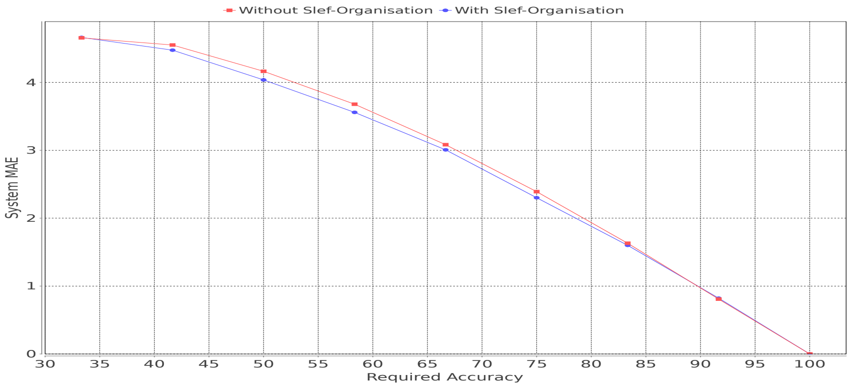


Fig. 4. MAE: OBRSP vs OBRSP with Self-Organisation

9 Conclusions and Future Work

In this paper we have shown how resource search can benefit from the creation of flexible organisations in a large distributed system where agents do not have a global view. To maintain the performance of the system, we have introduced a self-organising algorithm that keeps track of agents' connectivity and updates resources descriptions and availability. In contrast to the related work, our organisation formation algorithm groups agents with semantically close resources together instead of creating a coalition of agents to provide services. Subsequently, the search protocol would target specific organisation(s) which in turn improves the search performance. Our approach is also able to resolve the problem of resource availability; as there are many agents holding semantically close resources, there is a higher probability of finding the required resource within an organisation. Simulation results have shown an improved search performance when using the organisation-based resource search protocol by reducing the average number of hops a resource search message would need to find the required resources successfully. Currently, we are working on improving the organisation management mechanisms by which these organisations would self-adapt themselves as a response to changes in the environment or performance degradation.

Acknowledments. The work of Muntasir Al-Asfoor has been supported through a scholarship funded by the Iraqi Ministry of Higher Education and Scientic Research *(MOHESR)*.

References

1. Bernon, C., Chevrier, V., Hilaire, V., Marrow, P.: Applications of self-organising multi-agent systems: An initial framework for comparison. Informatica 30(1), 73–82 (2006)
2. Biskupski, B., Dowling, J., Sacha, J.: Properties and mechanisms of self-organizing manet and p2p systems. ACM Trans. Auton. Adapt. Syst. 2(1) (March 2007)

3. Bordini, R.H., Wooldridge, M., Hübner, J.F.: Programming Multi-Agent Systems in AgentSpeak using Jason (Wiley Series in Agent Technology). John Wiley & Sons (2007)
4. Cholvi, V., Felber, P., Biersack, E.: Efficient search in unstructured peer-to-peer networks. In: Proceedings of the Sixteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2004, pp. 271–272. ACM, New York (2004)
5. de Cerqueira Gatti, M.A., de Lucena, C.J.P.: A multi-environment multi-agent simulation framework for self-organizing systems. In: Di Tosto, G., Van Dyke Parunak, H. (eds.) MABS 2009. LNCS, vol. 5683, pp. 61–72. Springer, Heidelberg (2010)
6. Dimakopoulos, V.V., Pitoura, E.: Performance analysis of distributed search in open agent systems. In: Proceedings of the 17th International Symposium on Parallel and Distributed Processing, IPDPS 2003, p. 20.2. IEEE Computer Society, Washington, DC (2003)
7. Foster, I., Kesselman, C., Tuecke, S.: The anatomy of the grid: Enabling scalable virtual organizations. Int. J. High Perform. Comput. Appl. 15(3), 200–222 (2001)
8. Gjoka, M., Kurant, M., Markopoulou, A.: 2.5k-graphs: from sampling to generation. CoRR, abs/1208.3667 (2012)
9. Jacyno, M., Bullock, S., Payne, T., Luck, M.: Understanding decentralised control of resource allocation in a minimal multi-agent system. In: Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2007, pp. 208:1–208:3. ACM, New York (2007)
10. Klyne, G., Carroll, J.J.: Resource description framework (rdf): Concepts and abstract syntax (February 2004)
11. Ljubešić, N., Boras, D., Bakarić, N., Njavro, J.: Comparing measures of semantic similarity. In: Lužar-Stiffler, V., Dobrić, V.H., Bekić, Z. (eds.) Proceedings of the 30th International Conference on Information Technology Interfaces, Zagreb2, pp. 675–682. SRCE University Computing Centre (2008)
12. Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., Balan, G.: Mason: A multiagent simulation environment. Simulation 81(7), 517–527 (2005)
13. McGinnis, J., Stathis, K., Toni, F.: A formal framework of virtual organisations as agent societies. In: FAVO, pp. 1–14 (2009)
14. Minar, N., Burkhart, R., Langton, C., Askenazi, M.: The swarm simulation system, a toolkit for building multi-agent simulations (1996)
15. Neville, B., Pitt, J.: PRESAGE: A programming environment for the simulation of agent societies. In: Hindriks, K.V., Pokahr, A., Sardina, S. (eds.) ProMAS 2008. LNCS, vol. 5442, pp. 88–103. Springer, Heidelberg (2009)
16. Saha, G.K.: Web ontology language (owl) and semantic web. Ubiquity 2007, 1:1 (September 2007)
17. Justin Samuel, S., Sasipraba, T.: Article: Trends and issues in integrating enterprises and other associated systems using web services. International Journal of Computer Applications 1(12), 17–20 (2010)
18. Tisue, S., Wilensky, U.: Netlogo: A simple environment for modeling complexity. In: International Conference on Complex Systems, pp. 16–21 (2004)
19. Yang, B., Garcia-molina, H.: Efficient search in peer-to-peer networks (2002)

The 'Two-Hop Principle': Stability of MAS in Communications Networks

John Debenham[1] and Ante Prodan[2]

[1] QCIS, University of Technology, Sydney, Australia
john.debenham@uts.edu.au
[2] School of Computing & Mathematics, University of Western Sydney

Abstract. A joint university / industry collaboration is investigating the deployment of MAS technology to manage wireless communications networks. These systems are very large and are situated in a chaotic environment which suggests the use of distributed intelligent control that aims to continually improve, whilst not expecting to optimise, network performance. A key problem is ensuring the *stability* of these large, distributed control systems. A sequence of management systems have been developed over four years from which we have abstracted our 'two-hop principle'. We conjecture that the principle has wider application to multiagent network management systems. The generic principle is described and experiments are reported to illustrate the stability observed.

1 Introduction

This work follows from [1] in which we discussed the self-organisation of an 802.11 mesh network using a multiagent system. The core idea for that system has since been adapted and applied to other wireless communications network management problems in a joint project between The University of Technology, Sydney, The University of Western Sydney and Alcatel-Lucent Bell Labs (Paris). Following the success of these experiments we are now confident to propose the 'Two-Hop Principle' as a general approach to obtaining stability of multiagent systems applied to large, communications network management problems.

The massive scale and high degree of dynamism of communications networks dictates that their management can only be achieved by distributing intelligence to agents to key points in the network. This should ensure that the solution is scalable, and that it responds in a timely way. A key problem for distributed control is ensuring the *stability* of the algorithms, and that is addressed here.

In this collaborative work systems have been designed for 802.11 and LTE networks. 802.11 and LTE networks are fundamentally different. In 802.11 networks some of the nodes, or stations, are wired into the backbone and the remainder perform a relay function between mobile devices and the wired nodes. Two key parameters in these networks is the *topology* — i.e. which nodes communicate with each other in pairs so determining the number of hops on the path from mobile device to the wired network, and the *choice of channel* — there are eleven channels in the 802.11 standard with seven only non-overlapping (i.e. non-interfering).

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 281–292, 2013.

In the more recent LTE networks, such as the Alcatel-Lucent lightRadio™ cube technology, all nodes are wired and so there is no topology problem — the issue addressed for LTE networks is *power conservation*.

These applications were managed by multiagent systems. The basic elements of the model are: a set of *mobiles* (mobile devices), a network of *stations*, and intelligent *agents*. One agent is assigned to each station. In 802.11 networks the agents are physically embedded in each wireless device. In LTE networks the multiagent system resides in a cloud and the agents communicate with each other, and with their respective nodes using the wired network [2].

A *mobile* is an abstraction of a handheld device — the set of mobiles at any time[1] is denoted by $\mathcal{M}$. There are a random number of mobiles that we assume are somehow randomly distributed in a *service region* $\mathcal{S}$ that is considered as two-dimensional. The performance of a mobile is largely determined by the communication bandwidth that the network supplies. For every mobile $m \in \mathcal{M}$, β_{op} is the universally acceptable operational bandwidth.

Section 2 describes related work, and Section 3 defines an abstract model of a communications network. The multiagent systems, its associated logic including the two-hop principle, is presented in Section 4. This is followed by Section 5 that describes two of the experimental systems, and Section 6 concludes.

2 Related Work

Recent work on 802.11 Mesh Networks, such as [3], is predicated on a network whose prime purpose is to route traffic to and from nodes connected to the wired network — in which case there is assumed to be no direct traffic between end-user nodes. This introduces the conceptual simplification that mesh nodes can be seen as being grouped into clusters around a wired node where each cluster has a tree-like structure, rooted at a wired node, that supports the traffic. This is the prime purpose of 802.11 Mesh Networks in practice. Where possible, we move away from any assumptions concerning tree-like structures with the aim of designing algorithms for the more general classes of "wireless ad-hoc networks" or "wireless mesh networks". This work draws from previous work in the area of mesh networking [4], and in particular in distributed algorithms at Columbia University, Microsoft Research, University of Maryland and Georgia Institute of Technology. See also: [5], [6], [7] and [8].

There are three principal inputs that we assume are available to the methods described: a load model, a load-balancing algorithm and an interference model. The work described below makes no restrictions on these three inputs other than that they are available to every node. The load model, and so too the load balancing algorithm, will only be of value to a method for self-organisation if together they enable future load to be predicted with some certainty. We assume that the external demands on a set of nodes S are known and that there is a *load*

[1] The set of mobiles will alter in time, and so $\mathcal{M}^t$ is a more appropriate notation. To simplify the notation we generally omit the temporal t superscript from much of this work and, unless stated otherwise, all sets are taken to be "as at now".

balancing algorithm — that may or may not be intelligent — that determines how the load is routed through S. We assume that the load balancing algorithm will determine how the load is allocated to each link in the mesh.

Power conservation in LTE and other networks is being investigated in various 'green power' projects [9]. Information and communication technology (ICT) is responsible for 3% of the worldwide energy consumption, and contributes to 2% of the worldwide CO2 emission (Vadgama 2009). As one part of ICT sector, mobile radio networks contributes to a rather small portion of CO2 emissions (0.2%), but energy consumed is not negligible [10]. In mobile radio networks, over 50% of the total power is consumed by base stations [10], thus base stations can be targeted for reducing energy consumption.

The Alcatel-Lucent lightRadio^{TM} cube is LTE technology [11] claimed to reduce energy consumption of mobile networks by up to 50% over current technology. The lightRadio^{TM} cube is designed to complement or replace the existing large base stations that currently supply mobile signal. They employ multiple frequency bands and supports 2G, 3G and LTE networks. Due to its small size, the cube is very easy to instal. The cubes support beamforming in both the horizontal and vertical dimensions. Alcatel-Lucent claims a capacity improvement of 30% through vertical beamforming alone [12].

3 The Network

Each *station* $c \in \mathcal{C}$ transmits in a radial *beam* within which it delivers service to mobiles. A station c's beam is specified as a pair, $(\underline{a}(c), \overline{a}(c))$, where $\underline{a}(c), \overline{a}(c) \in [0°, 360°]$, $\underline{a}(c) < \overline{a}(c)$. The angle of c's beam is: $a(c) = \underline{a}(c) - \overline{a}(c) \leq a_{max}(c)°$ the greatest angle that the station is capable of — angles being measured with respect to an arbitrary, fixed standard direction. The stations are assumed to have on-board intelligence to manage their beam angle and direction. A 'stack' of stations is also assumed to intelligently coordinate the beam management of the stations in the stack.

Each station $c \in \mathcal{C}$ has range of transmission that is determined by the transmission power of a station, the angle of its beam, the distance-power gradient, and by any associated interference. The term *range of effective transmission*, $\rho(c)$, means that any mobiles connected to c receive at at least β_{op} bits/s/Hz when interference effects are ignored. That is, $\rho(c)$ will be determined by: the transmit power of c, c's beam angle, the distance-power gradient and the characteristics of c's antenna.

Given a station $c \in \mathcal{C}$ at some location $l(c)$, its beam $(\underline{a}(c), \overline{a}(c))$ and its range of effective transmission $\rho(c)$ determine a segment of a circle centred at c within which c may service mobiles. This segment consists of a set of points in two-dimensional space and is called the *segment* of c; it is specified by:

$$s(c_g) = (l(c), \rho(c), \underline{a}(c), \overline{a}(c))$$

$l(c)$ can be specified as Cartesian coordinates with respect to some arbitrary origin.[2] A station's *maximal segment* is:

$$s^*(c_g) = (l(c), \overline{\rho}(c), 0°, 360°)$$

where $\overline{\rho}(c)$ is the maximal range that c is capable of servicing with an isotropic antenna. We expect[3] $\rho(c) > \overline{\rho}(c)$.

The stations are somehow distributed in a finite region of two dimensional space $\mathcal{S}$ where they service a randomly varying number of mobiles that we assume somehow move randomly in that *service region* $\mathcal{S}$. For the network to be performing satisfactorily: $\forall m \in \mathcal{M}, \exists c \in \mathcal{C} \cdot b(m, c) > \beta_{op}$. If $\mathcal{C}_G \in \mathcal{C}_\mathcal{G}$ is a set of settings for the stations $\mathcal{C}$ such that $\mathcal{S} \subseteq \cup_{c \in \mathcal{C}} s(c_g)$ then the set of settings $\mathcal{C}_G$ for $\mathcal{C}$ is said to *cover* $\mathcal{S}$. It is assumed that: $\mathcal{S} \subseteq \cup_{c \in \mathcal{C}} s(c_g)$ at all times; otherwise additional stations are required to cover $\mathcal{S}$.[4]

A station c's parameter *settings*, c_g include its:

- beam angle $(\underline{a}(c), \overline{a}(c))$,
- transmit power[5] $P_t(c)$ (for the LTE and 802.11 systems)
- topology if it is linked to another station (802.11 systems)
- channel Γ_c (802.11 systems)

Any automatic adjustment of parameters should ideally ensure that the adjustment does not reduce the bandwidth at any mobile to less than β_{op}. This poses the problem of how a set of stations is to determine whether or not their combined segments deliver satisfactory bandwidth to the entire service region. That is, how are the stations to understand the topology of the service space?

This question can be addressed in a number of ways of which we mention two. The stations can be given:

1. a *region map*. The problem with a map is that it represents the objects in the region but not how those objects affect the propagation of signals through the region. For example, a 'wall' may either block a signal or permit it to partially penetrate it.
2. a finite set of "reference points" $\mathcal{R}$, $\mathcal{R} \subset \mathcal{S}$, in the service region. A *reference point* is a point $r \in \mathcal{R}$ is such that *if* $\forall c \in \mathcal{C}, r \in s(c_g)$ *then* c knows the sub-quadrant in which it lies. The idea being that there are sufficiently many reference points so that if station c adjusts its parameters then it will know the name of a reference point that it could previously service but can now no longer do so. The reference points are set up by the installing engineer as

[2] If c is servicing a region it means that all mobiles in that region can communicate with c; it does not necessarily mean that they will all be connected to c — they could be connected to other stations.

[3] The servicing range should be smaller than the maximal servicing range.

[4] For example, $a_{max}(c)$ may not be 360°. Stations should be of sufficient number, arranged in stacks or otherwise, to deliver complete coverage of their local region.

[5] In [1] we argued that it is more convenient to use $P_{\nu,\iota}(c)$ — the received power at the "close-in distance", ν, when c is using an isotropic antenna.

described below. The advantage of reference points over region maps is that they represent what actually happens whereas maps represent what could or should happen.

Reference points, maps, or some equivalent device, are necessary for a reliable solution. Without them a station can not understand the topology of the region that it is servicing.

If a station adjusts its parameters then this may well effect the desirable parameter settings of neighbouring stations. So stations need some understanding of the concept of 'neighbourhood'. The set of stations $\mathcal{C}_1(c)_g = \{c' \in \mathcal{C} \mid \exists r \in \mathcal{R}, r \in s(c_g) \wedge r \in s(c'_{g'})\}$ is the *one-hop set*[6] of c, and the set of stations $\mathcal{C}^*_1(c)_g = \{c' \in \mathcal{C} \mid \exists r \in \mathcal{R}, r \in s^*(c_g) \wedge r \in s^*(c'_{g'})\}$ is the *maximal one-hop set* of c. The set of stations $\mathcal{C}_2(c)_g = \{c' \in \mathcal{C} \mid \exists r \in \mathcal{R}, \exists c'' \in \mathcal{C}_1(c)_g, r \in s(c'_{g'}) \wedge r \in s(c''_{g''})\}$ is the *two-hop set* of c. In general, the set of stations $\mathcal{C}_{n+1}(c)_g = \{c' \in \mathcal{C} \mid \exists r \in \mathcal{R}, \exists c'' \in \mathcal{C}_n(c)_g, r \in s(c'_{g'}) \wedge r \in s(c''_{g''})\}$ is the $(n+1)$-*hop set* of c.[7] If every station has at least one reference point in its segment for all of its settings then: $c \in \mathcal{C}_1(c)$ and $\mathcal{C}_n(c) \subset \mathcal{C}_{n+1}(c)$.

When the network is installed we assume than an engineer will calibrate the reference points. To calibrate reference point r all stations within whose range r lies are set to include r in their beam (set at a standard angle) and transmit at a standard median power. Then for each of these stations in turn, they transmit on their maximum power setting and on their minimum power setting. For each of these two transmission settings the received power at r is recorded. In Section 5 we will use these recorded received power values to interpolate for a reference point r an estimate of the identity of the station that delivers the signal of greatest strength given the actual power settings of r's neighbouring stations.

4 Multiagent System

Each station $c \in \mathcal{C}$ is uniquely associated with an agent, $\gamma_c \in \mathcal{A}$. An agent is responsible for adjusting its station's parameter settings and for interacting with nearby agents. Each station's beam angle and direction is managed by its beam management algorithms that we assume are on-board as described in Section 3. For example, if a station's power is adjusted then we expect this to trigger adjustments to the station's beam.

Our autonomous agents are hybrid agents. Their interaction protocol is presented in Section 4.1 and their logic consists of two components:

– a *reactive component* that deals with unexpected problems — in our case this is limited to station overload when its mobiles' bandwidths fall below the critical level β_{op}. The reactive logic acts quickly, without consultation or deliberation. See Section 4.2

[6] Alternatively $C_1(c)$ could be called the *interference set* of c as it delineates the extent of c's interference.

[7] It may appear more correct to define the hop-sets using a point $y \in \mathcal{S}$ rather than a reference point $r \in \mathcal{R}$ but this would lead to computational difficulties as $\mathcal{S}$ is unbounded and $\mathcal{R}$ is finite.

– a *proactive component* that aims to achieve the system goal — in our case this is to move the settings towards G^* the optimal setting. See Section 4.3.

As is often the case for autonomous agents, the reactive logic overrides the proactive logic.

4.1 Agent Interaction

The only actions that software agents can make is to send and receive messages. To do this an agent needs an illocutionary communication language such as that described in [13]. Agent interaction is simplified for cooperative agents with a common goal and a common utility function. In this case an agent γ can propose a set of commitments whose enactment increases utility from its perspective knowing that this proposal is acceptable from the perspective of the other agents in the negotiation who will then cooperatively enact their commitments. This is how our deliberative autonomous agents cooperate in Section 4.3.

If an autonomous agent is to spontaneously propose to adjust a parameter setting of their station and other nearby stations, it is necessary to ensure that another proposal is not under way in the same region at the same time [14]. To manage this potential problem we adopt the following simple "locking" procedure. Suppose that agent γ_c for station c wishes to propose that the settings for $\mathcal{C}_1(c)$ be changed from: $\mathcal{C}_1(c)_G \in \mathcal{C}_1(c)_\mathcal{G}$ to: $\mathcal{C}_1(c)_{G'} \in \mathcal{C}_1(c)_\mathcal{G}$. Agent α_c communicates with the agents for stations $\mathcal{C}_2(c)$ as follows:

– send message "request lock by c" to all agents in $\mathcal{C}_2(c)$
– **if** all agents in $\mathcal{C}_2(c)$ respond "accept lock from c" **then**:
  • send message "propose change $\mathcal{C}_1(c)_G$ to: $\mathcal{C}_1(c)_{G'}$" to all agents in $\mathcal{C}_1(c)$
  • when all agents in $\mathcal{C}_1(c)$ respond "$\mathcal{C}_1(c)_G$ changed to: $\mathcal{C}_1(c)_{G'}$" then
    ∗ send message "request unlock by c" to all agents in $\mathcal{C}_2(c)$
    ∗ if all agents in $\mathcal{C}_2(c)$ respond "accept unlock from c" then done
– **else** abandon

All of this is typically performed very quickly.

4.2 Reactive Logic

A station is *overloaded* when it is at, or 'near', capacity. When a station is overloaded it attempts to shed some of its load. This is achieved by *shedding load*; i.e. reducing power in the LTE problem, and by dropping connections[8] in 802.11 [15]. Adjustments to the beam angle, which are dealt with by on-board logic, will alter the region of effective transmission of the station and so may increase interference as well as the level of overload.

If a station is shedding load then it may find that some reference points that were previously covered are no longer covered; we say that such reference points

[8] If no other station is available to take a dropped connection then the mobile will be disconnected.

have been *abandoned* by the action. The reactive logic repeats the following simple procedure: while station c with settings c_g is overloaded it sheds load by one click (in LTE it reduces power by one click [16], and in 802.11 it drops a mobile connection) *and* for each reference point that c will abandon by this action c sends an inform message to the agent of each station in the set: $\{\gamma : \gamma_c \in \mathcal{C}_1(c_g)\}$ — those agents may already be aware of the overload situation via their listening capability — in particular, if an agent in $\{\gamma : \gamma_c \in \mathcal{C}_1(c_g)\}$ is aware of an overload at c and is switched off then it switches on, and commences its 'warm up' cycle.

The reactive logic is expressed as: event-condition-action rules that for station c is:

if observe overload at c
  and $\exists c' \in C_1(c)$ c' not overloaded
then c reduces load by one click
  and inform all in $\{\gamma : \gamma_c \in \mathcal{C}_1(c_g)\}$ "overload at c"
if c' receives "overload at c", $c \in C_1(c')$
  and c able to increase coverage
then c increases coverage

4.3 The 'Two Hop' Principle

The 'two-hop' principle is:

> Agent γ_c for station c desires to change the settings of $\mathcal{C}_1(c)$ from: $\mathcal{C}_1(c)_G \in \mathcal{C}_1(c)_{\mathcal{G}}$ to: $\mathcal{C}_1(c)_{G''} \in \mathcal{C}_1(c)_{\mathcal{G}}$ if γ_c estimates that: $\boldsymbol{U}(\mathcal{C}_2(c)_{G''}) > \boldsymbol{U}(\mathcal{C}_2(c)_G)$.

The general idea is that agent γ_c will propose to adjust the parameters in $\mathcal{C}_1(c)$ as long as the proposal is expected to result in a net increase in utility for $\mathcal{C}_2(c)$. This is the 'trick' that appears to ensure[9] stability [17].

As the adjustment process occurs very quickly we assume that the state of the mobiles is unchanged during the process. The process then consists of two steps: first agent γ_c proposes to adjust the power for stations in $\mathcal{C}_1(c)$, and second the beam management algorithms adjust the stations' beams — when this is all complete the agents in $\mathcal{C}_1(c)$ confirm that the change is completed as described in Section 4.1. For this process to be (fairly) certain of yielding a net increase in utility for $\mathcal{C}_2(c)$: (1) the parameter adjustment step needs to be significant and, (2) the beam adjustment step needs to be 'intelligent' as we define below. In detail these two steps are:

1. γ_c proposes to adjust the parameter settings for stations in $\mathcal{C}_1(c)$ from $\mathcal{C}_1(c)_G$ to $\mathcal{C}_1(c)_{G'}$ if:

$$\gamma_c \text{ estimates that } \boldsymbol{U}(\mathcal{C}_2(c)_{G''}) > \boldsymbol{U}(\mathcal{C}_2(c)_G), \text{ and}$$
$$\boldsymbol{U}C_1(c)_G > \boldsymbol{U}(C_1(c)_{G'} \times \kappa$$

   where κ is a constant, $\kappa > 1$, to ensure that the parameter adjustment is significant, and

[9] Unfortunately we have been unable to demonstrate this analytically.

2. the beam management algorithms do their stuff and change the station settings from $\mathcal{C}_2(c)_{G'}$ to: $\mathcal{C}_2(c)_{G''}$.

This will lead to a net increase in utility for $\mathcal{C}_2(c)$ delivered bandwidth is not compromised: $||\{m \in \mathcal{M} : \forall c \in \mathcal{C} \cdot b(m,c) < \beta_{op}\}||$ is not increased — see [1]. We hypothesise that this should occur if: $I(\mathcal{C}_2(c)_{G'}) \leq I(\mathcal{C}_2(c)_{G''})$ which we conveniently adopt as the meaning of *intelligent beam forming.*

5 Two Systems for Adjusting the Power: LTE/802.11

As noted in Section 1 the management of LTE and 802.11 networks are fundamentally different due to the networking of the managing agents in LTE. The work described in focusses on the power adjustment problem in an 802.11 system and in an LTE system. The second system benefitted from preliminary testing with a NetLogo simulator — see Figure 1. Formalising the *power conservation problem*, given a set of stations $\mathcal{C}$ we define a two-dimensional *utility function* across the space of the settings of $\mathcal{C}$, $\mathcal{C}_{\mathcal{G}}$. $\boldsymbol{U} : \mathcal{C}_{\mathcal{G}} \to \mathbb{R}^2_+$, the two-dimensional space of positive reals. Given a set of settings $\mathcal{C}_G$ for a set of stations $\mathcal{C}$, the first component of $\boldsymbol{U}$: $U_1(\mathcal{C}_G) \triangleq \sum_{c\in\mathcal{C}} P_{\nu.\iota}(c)$, and the second component: $U_2(\mathcal{C}_G) \triangleq I(\mathcal{C}_G)$, where $I(\mathcal{C}_G)$ is an estimate of the interference between the stations in $\mathcal{C}$ in setting $\mathcal{C}_G$ — see [1] for an estimate of $I(\mathcal{C}_G)$. An *ordering* is defined for $\boldsymbol{U}$ as follows: suppose $\boldsymbol{U}(\mathcal{C}_G) = (p, i)$ and $\boldsymbol{U}(\mathcal{C}_{G'}) = (p', i')$ then: $p < p' \to \boldsymbol{U}(\mathcal{C}_G) > \boldsymbol{U}(\mathcal{C}_{G'})$, and $(p = p') \wedge (i < i') \to \boldsymbol{U}(\mathcal{C}_G) > \boldsymbol{U}(\mathcal{C}_{G'})$. This ordering of $\mathbb{R}^2_+$ gives first priority to saving power, and second priority to reducing interference. These two priorities are interlinked in that reducing interference, perhaps by narrowing a beam angle, may then reveal new opportunities for power saving. Further adjusting power may then lead of adjustment of the beams.

The *power conservation problem* is: given a set of stations $\mathcal{C}$, and a set of mobiles $\mathcal{M}$, to find the settings $\mathcal{C}_{G^*}$ for $\mathcal{C}$:

$$\begin{aligned}\mathcal{C}_{G^*} = \arg\max_{\mathcal{C}_G} \{&\boldsymbol{U}(\mathcal{C}_G) : \\ &||\{m \in \mathcal{M} : \forall c \in \mathcal{C} \cdot b(m,c) < \beta_{op}\}|| \text{ is minimal}\}\end{aligned}$$

where $||\{m \in \mathcal{M} : \forall c \in \mathcal{C} \cdot b(m,c) < \beta_{op}\}||$ is the number of mobiles that are receiving less than β_{op} — hopefully this will be zero. In other words, for all settings that deliver satisfactory bandwidth, to find the setting with highest utility.

The criterion for a satisfactory solution,

$$||\{m \in \mathcal{M} : \forall c \in \mathcal{C} \cdot b(m,c) < \beta_{op}\}|| \text{ is minimal}$$

is but one suitable criterion.

It remains to discuss how γ_c will determine G' that starts the proactive process in the first step above [18]. If station c reduces its transmit power then it will in general abandon references points at which it delivers the strongest signal. So c needs to know which reference points will be abandoned if its transmit power

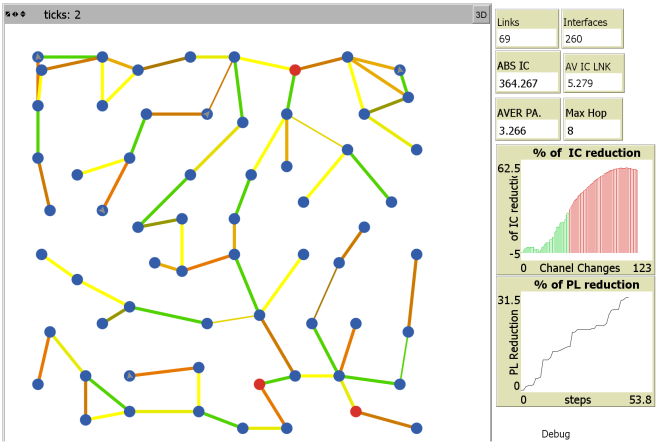


Fig. 1. The NetLogo simulator used for preliminary testing

is reduced by some amount. It uses the reference point calibration settings to do this — described in Section 3. Suppose reference point r is in the range of station c's beam. c accesses the data for r and identifies all of r's neighbouring stations including c. For each of these neighbours, c', c interpolates an estimate of c''s received power at r given c''s actual transmit power, $P_t(c')$, or $P_{\nu,\iota}(c')$. c then identifies the c' for which the received power at r is greatest.

Station c now knows the set of reference points at which it delivers maximum power. Mobiles at those reference points will connect to c given that c has available capacity. Station c also knows how this set of reference points will change if it, or any of r's neighbours, change their power settings. Now stations are connected to mobiles and not to reference points. The number of mobiles connected to any station is common knowledge, but we assume that their precise location is known to no one. We deal with this lack of knowledge as follows: we assume that the mobiles connected to a station are somehow randomly distributed over the reference points to which it delivers the strongest signal. That is, we assume that those mobiles are "bunched together in clumps" at one of those reference points. For simplicity we assume that the distribution of mobiles over reference points is uniform.

So c now knows which reference points will be abandoned if it reduces power and the load (i.e. number of mobiles) associated with those reference points. c also knows which stations could service those potentially abandoned reference points, and how much additional power they would use to do so. And, given that another station may have to increase power to service an abandoned reference point it may mean that yet another station can reduce power. At a simplistic level then the problem of determining G' is a large search problem through all these possibilities. One way to reduce this search space is to limit the possibilities to those associated with the the least number of abandoned reference points possible — i.e. consider only the smallest reduction of load that abandons reference points. This makes good practical sense as making significant changes to the settings when the set of mobiles is changing could lead to instability.

5.1 Results and Discussion

The results from simulation experiments on the power conservation system are heavily dependent on the simulation data. When the demand data is constant and gradually increasing or decreasing the power conservation is near optimal — including 'night time' scenes when the load is very low. As the demand simulation data becomes more erratic the saving is reduced particularly when stations are switched off due to the warm-up time of over one minute — under such circumstances an unexpected flood of demand will simply overload the system. When the demand is such that it requires all stations to be switched on then the saving over the maximum power setting increases as the load decreases with a ball-park mean of around 40% saving as demand drops from a maximal level to a level at which stations will turn off.

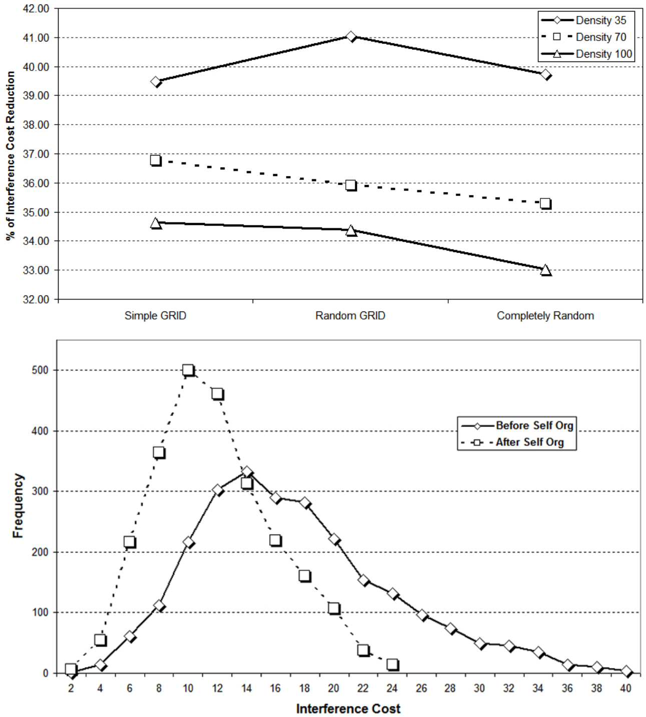


Fig. 2. (a) Interference cost reduction as a function of topologies. (b) Comparison of *IC* across the network before (taller graph) and after (flatter graph) self-organisation.

Impact of Typical Topologies on the Interference Cost. Figure 2(a) shows the variation in the interference cost reduction as a function of network topology across different node densities. It can be deduced that the impact of the topologies on the performance of the algorithm (i.e. in terms of interference cost reduction) is insignificant. The mean of IC reduction calculated from the data obtained shows that the topology with the smallest average IC reduction is the completely random with a mean of 36.02 and topology with the most IC reduction is the random grid with a mean of 37.12. The difference in performance between best and worst case is just 1.1 which confirms that the performance of the algorithm is fairly independent of the type of topology.

Performance Comparison Across the Network. In this study, we obtained interference cost (IC) in different regions of the MR-WMN (Multi Radio Wireless Mesh Network) for the same set of links before and after the self-organisation algorithm is invoked. Comparison of the results obtained is shown in Figure 2(b) where the Interference cost is on the X-axis. From Figure 2(b) we can see that there were no nodes (the 'peaked' graph) that caused more interference after the self-organisation than it had caused before (the flatter graph) the self-organisation was invoked.

6 Conclusion and Future Work

The 'two-hop' principle is formally defined in Section 4.3. It evolved and was abstracted from our experiments over four years in building distributed solutions to wireless telecommunications network management problems. Some of these assumptions were made to simplify the theoretical development and some were *ad hoc*. We have described simulations that are based on sweeping assumptions concerning the operation of the station's beam management algorithms for the LTE system [19], particularly the beam management algorithms are assumed to be "intelligent" in the sense described in Section 4.3 — we believe that this assumption is not unduly restrictive. We have also assumed that there is a load balancing algorithm in the two 802.11 systems — this is perfectly reasonable. Future work is addressing predictive systems for demand. This is presently being treated as a distributed data mining exercise coupled with a background mining system that looks for overall patterns in demand — such as a high demand near football fields on Saturdays in winter. We believe that the "two-hop principle" that lies at the heart of the deliberative logic is a sound way to manage the stability of these distributed control systems, and that it can be applied similarly to related problems in the management of wireless network communication, or indeed to any network management problem with chaotic demand for which in only makes sense to try to improve performance rather than optimise it.

Acknowledgement. This work is supported by an ARC Linkage Grant "Smart communications network management" [2008–2012] funded jointly by the Australian Research Council and Alcatel-Lucent (Bell Labs) Paris.

References

1. Debenham, J., Prodan, A.: A multiagent system for self-organisation of an 802.11 mesh network. In: Mařík, V., Strasser, T., Zoitl, A. (eds.) HoloMAS 2009. LNCS, vol. 5696, pp. 244–253. Springer, Heidelberg (2009)
2. 3GPP: Technical specification 25.214. physical layer procedures (fdd). Technical report (2005)
3. Li, T., Leith, D., Badarla, V., Malone, D., Cao, Q.: Achieving end-to-end fairness in 802.11e based wireless multi-hop mesh networks without coordination. Mobile Networks and Applications 16(1), 17–34 (2011)
4. Debenham, J., Prodan, A.: Self-organisation of an 802.11 mesh network. In: García-Pedrajas, N., Herrera, F., Fyfe, C., Benítez, J.M., Ali, M. (eds.) IEA/AIE 2010, Part I. LNCS, vol. 6096, pp. 113–122. Springer, Heidelberg (2010)
5. Ko, B.J., Misra, V., Padhye, J., Rubenstein, D.: Distributed Channel Assignment in Multi-Radio 802.11 Mesh Networks. Technical report, Columbia University (2006)
6. Mishra, A., Rozner, E., Banerjee, S., Arbaugh, W.: Exploiting partially overlapping channels in wireless networks: Turning a peril into an advantage. In: ACM/USENIX Internet Measurement Conference (2005)
7. Mishra, A., Shrivastava, V., Banerjee, S.: Partially Overlapped Channels Not Considered Harmful. In: SIGMetrics/Performance (2006)
8. Hiertz, G., Denteneer, D., Stibor, L., Zang, Y., Costa, X., Walke, B.: The IEEE 802.11 universe. IEEE Communications Magazine 48(1), 62–70 (2010)
9. Vadgama, S., Hunukumbure, M.: Trends in green wireless access networks. In: 2011 IEEE International Conference on Communications Workshops (ICC), pp. 1–5 (2011)
10. Lorincz, J., Capone, A., Bogarelli, M.: Energy savings in wireless access networks through optimized network management. In: Proceedings of the 5th IEEE International Conference on Wireless Pervasive Computing, Piscataway, NJ, USA, pp. 449–454. IEEE Press (2010)
11. Dahlman, E., Parkvall, S., Sköld, J., Beming, P.: 3G Evolution: HSPA and LTE for Mobile Broadband. Academic Press (2008)
12. Fehske, A.J., Richter, F., Fettweis, G.P.: Energy efficiency improvements through micro sites in cellular mobile radio networks. In: Globecom Workshops (2009)
13. FIPA: Communicative act library specification. Technical Report SC00037J, Foundation for Intelligent Physical Agents, Geneva, Switzerland (2002)
14. Karl, H., Willig, A.: Protocols and Architectures for Wireless Sensor Networks. John Wiley and Sons (2005)
15. Chieochan, S., Hossain, E., Diamond, J.: Channel assignment schemes for infrastructure-based 802.11 WLANs: A survey. IEEE Communications Surveys Tutorials 12(1), 124–136 (2010)
16. Ramachandran, K., Belding, E., Almeroth, K., Buddhikot, M.: Interference-aware channel assignment in multi-radio wireless mesh networks. In: Proceedings of Infocom 2006, Barcelona, Spain, pp. 1–12 (2006)
17. Prodan, A.: A Multiagent System for Topology Control on Multi-Radio Wireless Mesh Networks. PhD thesis, University of Technology, Sydney (2010)
18. Debenham, J., Prodan, A., Lin, K.: LTE power conservation. Technical report, UTS — Alcatel-Lucent (July 2012)
19. Fodor, G., Koutsimanis, C., Rácz, A., Reider, N., Simonsson, A., Müller, W.: Intercell Interference Coordination in OFDMA Networks and in the 3GPP Long Term Evolution System. Journal of Communications 4(7), 445–453 (2009)

Agent Interaction Protocols in Support of Cloud Services Composition

Djamel Benmerzoug, Mohamed Gharzouli, and Mounira Zerari

Faculty of New Technologies of Information and Communication,
University Constantine 2, Algeria
{benmerzoug,gharzouli}@gmail.com, zerari.mounirabis@yahoo.fr

Abstract. Combining different independent cloud services necessitates a uniform description format that facilitates the design, customization, and composition. In this context, Interaction Protocols (IP) are a useful way for structuring communicative interaction among business partners, by organizing messages into relevant contexts and providing a common guide to the all parts. The challenge here is twofold. First, we must provide a formal model that is rich enough to capture interactions characteristics. Second, we must allow designers to combine existing protocols to achieve a new specific need.

The paper introduces a formal analysis framework allowing the composition of IP. This framework is based on our previous work [7][8]. In this paper, we mainly focus on the compositions of IP, where particular protocols may then be selected and composed to support a desired business application.

1 Introduction

Business processes are becoming increasingly complex and integrated both within internal corporate business functions (e.g., manufacturing, design engineering, sales and marketing, and enterprise services) and across the external supply chain [24]. Cloud service offerings provide a competitive advantages to enterprises through flexible and scalable access to computing resources.

With the recent advances in cloud computing, there is a clear need for advanced business applications to coordinate multiple cloud services into a multistep business transaction.

Every new cloud service provider have their own way on how a user or cloud application interacts with their cloud leading to cloud API propagation [21]. Therefore, cloud adoption will be hampered if there is not a good way of integrating data and services across clouds. In fact, combining different independent cloud services necessitates a uniform description format that facilitates the design, customization, and composition.

Interaction Protocols (IP), which is inspired by work of Multiagent systems (MAS), are a useful way for structuring communicative interaction among business partners, by organising activities into relevant contexts and providing a

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 293–304, 2013.

common guide to the all parts. This approach requires distilling from the structure of a business collaboration the key capabilities that must necessarily be present in a business transaction and specifying them accurately and independently of any specific implementation mechanisms.

Because protocols address different business goals, they often need to be composed to be put to good use. For example, a process for purchasing goods may involve protocols for ordering, shipping, and paying for goods. Therefore, we would like to treat protocols as modular components, potentially composed into more complex protocols, and applied in a variety of business processes.

In this paper, we propose a basis for a theoretical approach for aggregating protocols to create a new desired business application. The proposed approach provides the underpinnings of aggregation abstractions for protocols.

The remainder of the paper is organized as follows: Section 2 overviews some related work. Section 3 presents our previous work : The use of IP to define and manage collaborative processes in B2B relationships. Based on that, in section 4, we define the concept of the cloud Business protocols. Section 5 gives details about protocols composition. and Section 6 provides some concluding remarks.

2 Related Work

Over the last few years, the prosperous research on collaborative enterprises applications has led to a multitude of results. In this section, we overview major techniques that are most closely related to our approach.

Web Services Based Approaches

With the growing trends of service oriented computing, composition of Web services has received much interest to support flexible inter-enterprise application integration. As stated in [17] and [25], current efforts in Web services composition can be generally grouped into three categories: manual, automatic, and semi-automatic composition.

By manual composition, we mean that the composite service is designed by a human designer (i.e., service provider) and the whole service composition takes place during the design time. This approach works fine as long as the service environment, business partners, and component services do not or rarely change. On the other hand, automatic service composition approaches typically exploit the Semantic Web and artificial intelligence planning techniques. By giving a set of component services and a specified requirement (e.g., user's request), a composite service specification can be generated automatically [9].

However, realizing a fully automatic service composition is still very difficult and presents several open issues [17],[9], [11]. The basic weakness of most research efforts proposed so far is that Web services do not share a full understanding of their semantics, which largely affects the automatic selection of services.

There exist some research efforts that encourage manual and automatic compositions[20][18]. Instead of coupling component services tightly in the service model, such approaches feature a high-level abstraction (e.g., UML activity model, protocol specifications, and interface) of the process models at the design time, while the concrete composite services are either generated automatically using tools or decided dynamically at run time (e.g., BPEL4WS [16]).

Our proposition is similar to these approaches in the sense that we also adopt a semi-automatic approach for application integration. The collaboration scenario is specified as interaction protocols not high-level goals and the component applications are selected, at run time, based on the protocol specification specified at the design time.

Also, the Web services related standards for services composition and interoperability, such as the BPEL4WS are lower level abstractions than ours since they specify flows in terms of message sequences. Also, they mix interaction activities and business logic making them unsuitable for reuse. In contrast to our approach, the BPEL4WS elements are only used to specify messages exchanges between the different business partners. Afterwards, this specification is used by agents to enact the integration of business processes at run time. Agents have the capability to dynamically form social structures through which they share commitments to the common goal. The individual agents, through their coordinated interactions achieve globally coherent behavior; they act as a collective entity known as a multiagent system. In our previous work [6][3], we have explored the relationship between web services, multiagent systems and enterprise application integration.

Agent Based Approaches

Multiagent systems are a very active area of research and development. In fact, several researchers are working at the intersection of agents and enterprise systems.

For example, Buhler et al. [12] summarize the relationship between agents and Web services with the aphorism Adaptive Workflow Engines = Web Services + Agents: namely, Web services provide the computational resources and agents provide the coordination framework. They propose the use of the BPEL4WS language as a specification language for expressing the initial social order of the multi-agent system. [12] does not provide any design issues to ensure the correctness of their interaction protocols.

Driven by the motivation for reuse of interaction protocols, [26] and [15] consider protocols as a modular abstractions that capture patterns of interaction among agents. In these approaches, composite protocols can be specified with a Protocol Description Language (such as: CPDL[1] or MAD-P[2]). Although formal, [26] and [15] do not provide any step for the verification of the composite protocols.

Agent-oriented software methodologies aim to apply software engineering principles in the agent context e.g. Tropos, AMCIS, Amoeba, and Gaia. Tropos [10][22] and AMCIS [5][4] differ from these in that they include an early requirements stage in the process. Amoeba [14] is a methodology for business processes that is based on business protocols. Protocols capture the business meaning of

[1] CPDL: Communication Protocol Description Language.

[2] MAD-P: Modular Action Description for Protocols.

interactions among autonomous parties via commitments. Gaia [13] differs from others in that it describes roles in the software system being developed and identifies processes in which they are involved as well as safety and liveness conditions for the processes. It incorporates protocols under the interaction model and can be used with commitment protocols.

Our methodology differs from these in that it is aimed at achieving protocol reusability by separation of protocols and business rules. It advocates and enables reuse of protocols as building blocks of business processes. Protocols can be composed and refined to yield more robust protocols.

3 Interaction Protocols in Support of Service-Based Enterprise Application

Our approach is based on the notion IP through which we specify complex services and break the complexity of running systems by recognising larger chunks that have a meaning in the application domain. This notion of IP, which is inspired by work of Multiagent systems (MAS), supports the modelling of composite services as entities whose business logic involves a number of interactions among more elementary service components.

Figure 1 shows our conceptual model for the treatment of service-based enterprise application. Business partners participating in this scenario publish and implement a public process. So, a public business process is the aggregation of the private processes and/or Web services participating in it.

Let us notice that private processes are considered as the set of processes of the enterprise itself and they are managed in an autonomous way to serve local needs. Whereas the public processes span organizational boundaries. They belong to the enterprises involved in a B2B relationship and have to be agreed and jointly managed by the partners.

Interaction protocols have been used in the area of multi-agent systems to represent interactions among agents. In the context of B2B relationships, a business protocol is a specification of the allowed interactions between two or more participant business partners. These interactions represent public business processes that enterprises agreed on the collaboration.

In our case the public process is specified by interaction protocols, which are used by the integrator agent to enact the public process at run time. Interaction protocols should be published, shared and reused among the business partners (see [3] for more detail).

3.1 Specification of Interaction Protocols

The B2B integration scenarios typically involve distributed business processes that are autonomous to some degree, hence the importance of Interaction protocols based modelling. They are a useful way for structuring communicative interaction among business partners, by organizing messages into relevant contexts and providing a common guide to the all parts. Formally an interaction protocol is defined as follow:

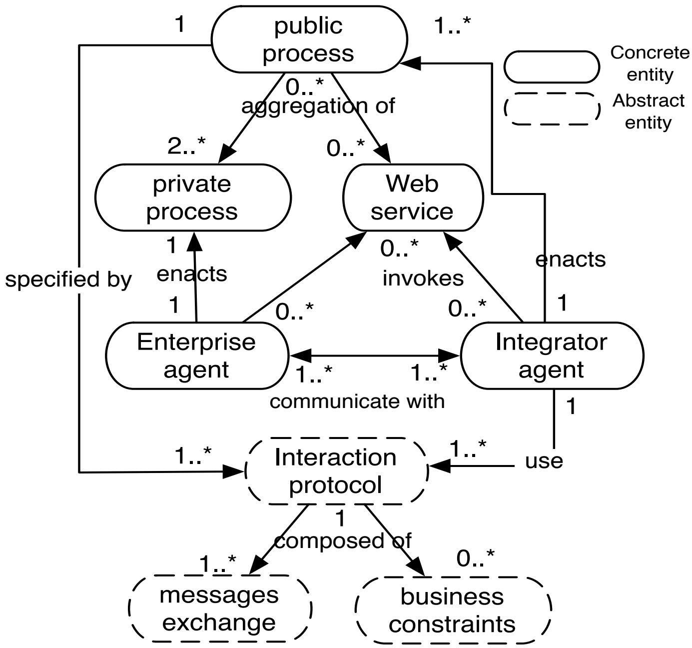


Fig. 1. Conceptual Model: Enterprise Application Based on Interaction Protocols

Definition 1. An Interaction Protocol is a quadruplet:

$$\text{IP} = <\text{ID, R, M, } f_M >\text{, where:}$$

- ID is the identifier of the interaction protocol
- $R = r_1, r_2, \ldots, r_n$ $(n > 1)$ is a set of Roles (private business process or Web Services)
- M is a set of non-empty primitive (or/and) complex messages, where:
  - A Primitive Message (PM) corresponds to the simple message, it is defined as follow:
    PM = <Sender, Receiver, CA, Option>, where:
    - Sender, Receiver $\in$ R
    - CA $\in$ FIPA ACL Communicative Act (such as: cfp, inform, ...)
    - Option: contain additional information (Synchronous / Asynchronous message, constraints on message,...)
  - A Complex Message (CM) is built from simpler (primitive) ones by means of operators:
    CM = $PM_1 op PM_2 \ldots op PM_m$, where:
    - $m > 1$, op $\in$ {XOR, OR, AND}, and
    - $\forall$ i $\in$ [1,m-1], $PM_i.Sender = PM_{i+1}.Sender$, $PM_i.Sender \in R$.
- f_M: a flow relation defined as : $f_M \subseteq$ (RxR), where (RxR) is a Cartesian product $(r_1, r_2) \in$ (RxR), for $r_1, r_2 \in$ R

Developing effective protocols to be executed by autonomous partners is challenging. Similar to protocols in traditional systems, IP in open and web-based settings need to be specified rigorously so that business partners can interact successfully.

For this reason, we have developed a method for IP design and verification. The proposed method (see figure 2) uses different models and languages.

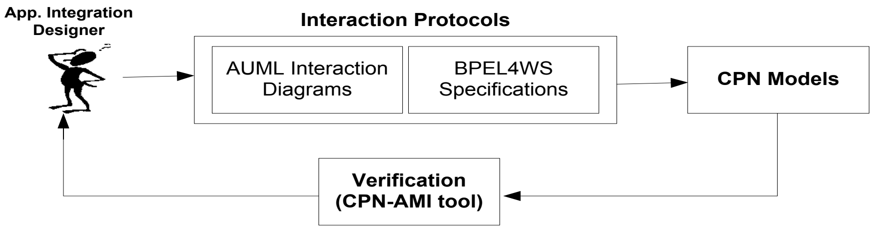


Fig. 2. The proposed method

Our method motivates the use of IP based on AUML/BPEL4WS, where pre- and post-conditions, rules, guards are specified in OCL[3].

AUML (Agent UML) notation [1][2] is a UML profile dedicated to agents trying to simplify the transition from software engineering to multi-agent system engineering. In other hand, BPEL4WS [16] (Business Process Execution Language for Web Services) is a de facto standard for describing Web services composition. In our context, BPEL4WS was used as a specification language for expressing the interaction protocols of the multi-agent system [6].

We also elaborated translation rules from interaction protocols notations used in our approach into Colored Petri Nets (CPN). These rules are implemented in IP2CPN: the tool we developed to automatically generate Petri nets from protocols specifications. Resulting Petri nets can be analyzed by dedicated tools to detect errors as early as possible.

4 Cloud Computing and Enterprises Application Collaboration

Cloud computing infrastructures can allow enterprises to achieve more efficient use of their IT hardware and software investments. They do this by breaking down the physical barriers inherent in isolated systems, and automating the management of the group of systems as a single entity. Cloud computing is an example of an ultimately virtualized system, and a natural evolution for data centers that employ automated systems management, workload balancing, and virtualization technologies.

[3] OCL: Object Constraint Language. (`www.omg.org/cgi-bin`).

4.1 Cloud Computing Models

There are various models and ways in which cloud computing can benefit an enterprise.

According to the intended access methods and availability of cloud computing environments, there are different models of deployment [23]. They include private cloud, public cloud, community cloud, and hybrid cloud, which are briefly analyzed below.

private cloud: In this model, the cloud infrastructure is exclusively used by a specific organization. The cloud may be local or remote, and managed by the enterprise itself or by a third party.

public cloud: Infrastructure is made available to the public at large and can be accessed by any user that knows the service location. In this model, no access restrictions can be applied and no authorization and authentication techniques can be used.

community cloud: Several organizations may share the cloud services. These services are supported by a specific community with similar interests such as mission, security requirements and policies, or considerations about flexibility.

hybrid cloud: Involves the composition of two or more clouds. These can be private, community or public clouds which are linked by a proprietary or standard technology that provides portability of data and applications among the composing clouds.

According to the Forrester Research market, many businesses want interoperability between their internal infrastructure combined with public cloud resources. They might want to use private application to process data in the cloud, they might want to use a cloud-based application to process private data, or they might want to use applications or tools that will run both private and on the public cloud. Consequently, we believe that a hybrid approach makes more sense for enterprises. In such approach, there is a need for complex developed business applications on the clouds to be interoperable. Cloud adoption will be hampered if there is not a good way of integrating data and applications across clouds.

In the next section, we introduce the Cloud Business Protocols, which are a useful way for structuring interaction among cloud resources, by organising activities into relevant contexts and providing a common guide to the all parts.

4.2 The Cloud Business Protocol

As stated in [19], to ensure a meaningful composition of two or more clouds, there is a clear need for placing emphasis on how to develop enhanced composite service offerings at the application-level and assign or reassign different virtual and physical resources dynamically and elastically. In fact, combining different independent cloud services necessitates a uniform description format that facilitates the design, customization, and composition.

Business protocol is a specification of the allowed interactions between two or more participant business partners. Applied to cloud computing, we propose the following variation: A Cloud Business Protocol is two or more business parties linked by the provision of cloud services and related information.

In fact, business parties in the cloud computing area are interconnected by the cloud business protocol. These parties are involved in the end-to-end provision of products and services from the cloud service provider for end cloud customers. Because protocols address different business goals, they often need to be composed to be put to good use. For example, a process for purchasing goods may involve protocols for ordering, shipping, and paying for goods.

Driven by the motivation of reuse, we would like to treat protocols as modular components, potentially composed into additional protocols, and applied in a variety of business processes. By maintaining repositories of commonly used, generic, and modular protocols, we can facilitate the reuse of a variety of well-defined, well-understood, and validated protocols.

In the rest of this paper, we propose a basis for a theoretical approach for aggregating protocols to create a new desired business application. The proposed approach provides the underpinnings of aggregation abstractions for protocols.

5 Business Protocols Composition

The composition of two or more business protocols generates a new protocol providing both the original individual behavioral logic and a new collaborative behavior for carrying out a new composite task. This means that existing protocols are able to cooperate although the cooperation was not designed in advance.

Definition 2. (Composite Protocol) A Composite Protocol (CP) is a tuple $CP = (P, Op, Input, Output, P_{init}, P_{fin})$ where:

- P is a non empty set of *basic protocols*,
- Op is a non empty set of *operators*, $Op \subseteq (P \times P)$,
- $Input, Output$ are a set of the elements required (produced) by the composite protocol CP,
- P_{init} is non empty set of *initial* protocols, $P_{init} \in P$ and
- P_{fin} is non empty set of *final* protocols, $P_{fin} \in P$.

The proposed approach provides the underpinnings of aggregation abstractions for protocols. To achieve this goal, we require an agreement between business protocols in the form of a shared contract.

5.1 Protocol Contract

For a correct composition of protocols, they must come to an agreement or contract. A contract describes the details of a protocol (participants in the protocol, produced elements, required elements, constraints,...) in a way that meets the mutual understandings and expectations of two or more protocols. Introducing the contract notion gives us a mechanism that can be used to achieve a meaningful composition.

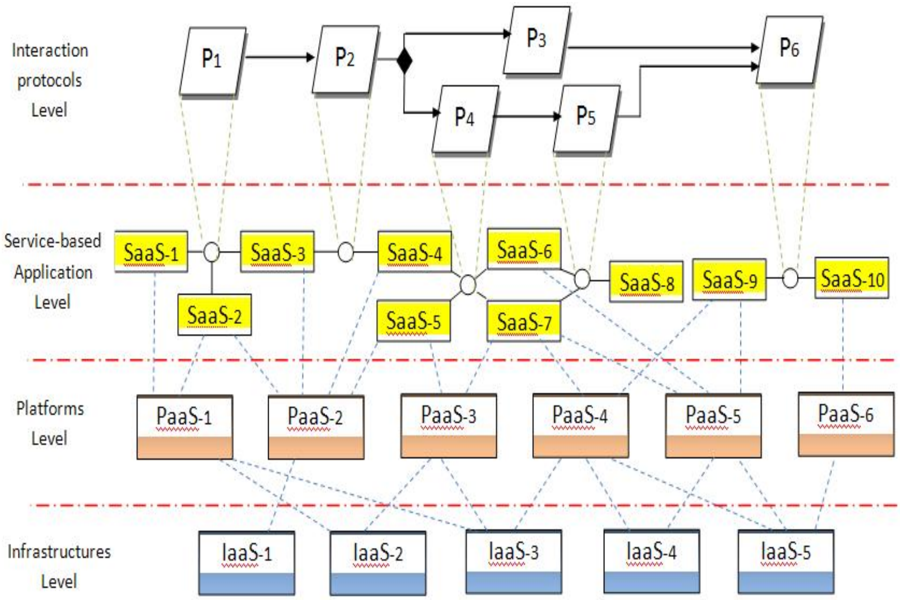


Fig. 3. An Example of Business Protocols Composition

Definition 3. (Protocol Contract) A contract C, is a collection of elements that are common across two protocols. It represents the mutually agreed upon protocol schema elements that are expected and offered by the two protocols.

1. Let us denote by P_k^{in}, P_k^{out} the set of elements required (produced) by the protocol P_k where $P_k^{in} = \{x_i, i \geq 1\}$ and $P_k^{out} = \{y_j, j \geq 1\}$
2. Let us define a function θ, called a *contract-mapping*, that maps a set of P_k^{out} elements (produced by the protocol P_k) and a set of P_r^{in} (consumed by the protocol P_r), $\theta = \big\{\exists\, y_i \in P_k^{out} \wedge \exists\, x_j \in P_r^{in} \mid (x_i, y_j)\big\}$, which means that the protocol P_r consumes the element y_j provided by the protocol P_k, and $C = (P_k^{out})_\theta(C) \cup (P_r^{in})_\theta(C)$.

5.2 Protocols Compositability Relationship

The compositability relationship means that business protocols can be joined together to create a new protocol that performs more sophisticated applications. That composition can then serve as a protocol itself. Figure 3 shows an example of business protocols composition where the composite protocol has six sub-protocols (called $P_1, P_2, \ldots, P_6$) and ♦ is a parallelism operator.

We can classify the nature of protocols composition on the dependance degree between them. We may thus distinguish between two kinds of Compositability relationship:

Definition 4. (Partial Compositability) Two protocols P_k and P_r meet the Partial Compositability relationship *iff* $\exists\ x_i \in (P_r^{in})_\theta(C), \exists\ y_j \in (P_k^{out})_\theta(C) \mid (x_i, y_j)$, which means that the protocol P_r can be executed after the protocol P_k but it must wait until all its "required elements" will be provided by other protocols.

In the example presented in figure 3, we have two partial Compositability relationships: (P_3, P_6) and (P_5, P_6).

Definition 5. (Full Compositability) Two protocols P_k and P_r are called Full Compositability *iff* $\forall x_i \in (P_r^{in})_\theta(C), \exists\ y_j \in (P_k^{out})_\theta(C) \mid (x_i, y_j)$, which means that the protocol P_r must be (immediately) executed after the protocol P_k because **all** its "required elements" are offered by the protocol P_k.

In our example, we have four full Compositability relationships: (P_1, P_2), (P_2, P_3), (P_2, P_4), and (P_4, P_5).

We note here that the partial (full) compositability relationship is not commutative. So, if a protocol P_k have a partial (full) compositability relationship with a protocol P_r, it does not means that P_r have a partial (full) compositability relationship with P_k .

Proposition. Let P $=\{P_1, P_2, \ldots, P_m\}$ a set of business protocols. The set P constitute a meaningful composition if:

$$\forall P_k \in (P - P_{init}\ ^4), \forall x_i \in (P_k^{in})_\theta(C), \exists\ y_j \in (P_r^{out})_\theta(C) \mid (x_i, y_j), \text{ where } r, k \in [1, m] \text{ and } r \neq k.$$

This proposition states that, if we have a set of protocols P $=\{P_1, P_2, \ldots, P_m\}$ where all their "required elements" are offered (by others protocols), this means that all the protocols of P can be executed.

6 Conclusion and Future Work

In this paper, we proposed a basis for a theoretical approach for aggregating protocols to create a new desired business application. The presented framework is based on our previous work [7][8] that provides a formal model for the composition of local participant implementations.

The key feature of our approach is the ability to re-use IP. It means that the IP is itself published and accessed as a Web service that can participate in other enterprise application collaboration. So, particular protocols may then be selected and composed to achieve a new specific need.

As future work, we will propose a set of operators that allows the creation of new value-added protocols using existing ones as building blocks. Sequence, alternative, iteration, and parallel are typical constructs specified in the control flow. We will also give a formal semantics to the proposed operators in terms of Petri nets.

[4] $P_{init} \in$ P and represent the initial protocols, which their "required elements" are provided by external events

References

1. Bauer, B., Bergenti, F., Massonet, P., Odell, J.J.: Agents and the UML: A Unified Notation for Agents and Multi-agent Systems? In: Wooldridge, M.J., Weiß, G., Ciancarini, P. (eds.) AOSE 2001. LNCS, vol. 2222, pp. 148–150. Springer, Heidelberg (2002)
2. Bauer, B., Odell, J.: UML 2.0 and agents: how to build agent-based systems with the new UML standard. International Journal of Eng. Appl. of AI 18(2), 141–157 (2005)
3. Benmerzoug, D.: Agent approach in support of enterprise application integration. International Journal of Computer Science and Telecommunications 4(1), 47–53 (2013)
4. Benmerzoug, D., Boufaida, M., Boufaida, Z.: Developing Cooperative Information Agent-Based Systems with the AMCIS Methodology. In: IEEE International Conference on Advances in Intelligent Systems: Theories and Application, Luxembourg. IEEE Press (November 2004)
5. Benmerzoug, D., Boufaida, Z., Boufaida, M.: From the Analysis of Cooperation Within Organizational Environments to the Design of Cooperative Information Systems: An Agent-Based Approach. In: Meersman, R., Tari, Z., Corsaro, A. (eds.) OTM Workshops 2004. LNCS, vol. 3292, pp. 495–506. Springer, Heidelberg (2004)
6. Benmerzoug, D., Boufaida, M., Kordon, F.: A Specification and Validation Approach for Business Process Integration based on Web Services and Agents. In: Proceedings of the 5th International Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems, MSVVEIS 2007, In conjunction with ICEIS 2007, pp. 163–168. NSTIIC Press (2007)
7. Benmerzoug, D., Kordon, F., Boufaida, M.: A Petri-Net Based Formalisation of Interaction Protocols Applied to Business Process Integration. In: Dietz, J.L.G., Albani, A., Barjis, J. (eds.) CIAO! 2008 and EOMAS 2008. LNBIP, vol. 10, pp. 78–92. Springer, Heidelberg (2008)
8. Benmerzoug, D., Kordon, F., Boufaida, M.: Formalisation and Verification of Interaction Protocols for Business Process Integration: a Petri net Approach. International Journal of Simulation and Process Modelling 4(3-4), 195–204 (2008)
9. Berardi, D., De Giacomo, D., Mecella, M., Calvanese, D.: Automatic composition of process-based web services: a challenge. In: Proc. 14th Int. World Wide Web Conf., WWW 2005 (2005)
10. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An Agent-Oriented Software Development Methodology. International Journal of Autonomous Agents and Multi-Agent Systems 8(3), 203–236 (2004)
11. Bronsted, J., Hansen, K.M., Ingstrup, M.: Service composition issues in pervasive computing. IEEE Pervasive Computing 9(1), 62–70 (2010)
12. Buhler, P.A., Vidal, J.M.: Towards adaptive workflow enactment using multiagent systems. International Journal on Information Technology and Management 6, 61–87 (2005)
13. Cernuzzi, L., Molesini, A., Omicini, A., Zambonelli, F.: Adaptable multi-agent systems: the case of the gaia methodology. International Journal of Software Engineering and Knowledge Engineering 21(4), 491–521 (2011)
14. Desai, N., Chopra, A.K., Singh, M.P.: Amoeba: A Methodology for Fodeling and Evolving Cross-Organizational Business Processes. Journal of ACM Trans. Softw. Eng. Methodol. 19(2) (2009)

15. Desai, N., Singh, M.P.: A modular action description language for protocol composition. In: Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, pp. 962–967. AAAI Press (2007)
16. SAP Siebel Systems IBM, Microsoft. Business process execution language for web services version 1.1. Technical report (2003)
17. Milanovic, N., Malek, M.: Current solutions for web service composition. IEEE Internet Computing 8(6), 51–59 (2004)
18. Montali, M., Pesic, M., van der Aalst, W.M.P., Chesani, F., Mello, P., Storari, S.: Declarative specification and verification of service choreographiess. International Journal of ACM Transactions on the Web 4(1) (2010)
19. Nguyen, D.K., Lelli, F., Papazoglou, M.P., van den Heuvel, W.-J.: Blueprinting Approach in Support of Cloud Computing. International Journal of Future Internet 4(1), 322–346 (2012)
20. Papazoglou, M.P., Pohl, K., Parkin, M., Metzger, A. (eds.): Service Research Challenges and Solutions. LNCS, vol. 6500. Springer, Heidelberg (2010)
21. Parameswaran, A.V., Chaddha, A.: Cloud interoperability and standardization. SETLabs Briefings 7(7), 19–26 (2009)
22. Penserini, L., Kuflik, T., Busetta, P., Bresciani, P.: Agent-based organizational structures for ambient intelligence scenarios. Ambient Intelligence and Smart Environments 2(4), 409–433 (2010)
23. Mell, P., Grance, T.: The NIST Definition of Cloud Computing (2009)
24. Dustdar, S., Papazoglou, P.M., Traverso, P., Leymann, F.: Service-oriented computing: a research roadmap. Int. Journal of Cooperative Information Systems 17(2), 223–255 (2008)
25. Sheng, Q.Z., Benatallah, B., Maamar, Z., Ngu, A.H.H.: Configurable composition and adaptive provisioning of web services. IEEE T. Services Computing 2(1), 34–49 (2009)
26. Vitteau, B., Huget, M.-P.: Modularity in interaction protocols. In: Dignum, F.P.M. (ed.) ACL 2003. LNCS (LNAI), vol. 2922, pp. 291–309. Springer, Heidelberg (2004)

Big Data Challenges in Industrial Automation

Marek Obitko[1], Václav Jirkovský[1,2], and Jan Bezdíček[1]

[1]Rockwell Automation Research Center,
Pekařská 695/10a, Prague, Czech Republic
[2] Department of Cybernetics, Faculty of Electrical Engineering,
Czech Technical University, Prague, Czech Republic
{mobitko,vjirkovsky,jbezdicek}@ra.rockwell.com

Abstract. Within the industrial domain including manufacturing a lot of various data is produced. For exploiting the data for lower level control as well as for the upper levels such as MES systems or virtual enterprises, the traditional business intelligence methods are becoming insufficient. At the same time, especially within internet companies, the Big Data paradigm is getting higher popularity due to the possibility of handling variety of large volume of quickly generated data, including their analysis and immediate actions. We discuss Big Data challenges in industrial automation domain, including describing and reviewing relevant applications and features. We pay special attention to the use of semantics and multi-agent systems. We also describe possible next steps for Big Data adoption within industrial automation and manufacturing.

Keywords: Big Data, large scale data, industrial automation, manufacturing, transportation, smart grid, internet of things, plant data, semantics, agents.

1 Introduction

Using large amount of data for better decision making is getting more possible with the technologies pioneered by web companies. The infrastructure was developed that is able to store clickstreams and user behavior that can be then used for example for offering relevant products and/or services. At the same time, similar or even greater amounts of data are coming from other domains, including for example manufacturing – the constant flow of data from various sensors creates huge amount of data. As an example, a CPG (Consumer Packaged Goods) company mentioned in [9] generates 5000 data samples every 33 milliseconds leading to 4 trillion of samples per year.

The impacts of data driven decision making processes were recognized in many areas and there is a growing enthusiasm for the notion of “Big Data”. This term by itself evokes a notion of large scale data, but more properties are contained in the research behind – the problems with variety and velocity need to be considered as well. The traditional approaches as used with relational databases and classical business intelligence methods are not sufficient for the analysis required for the Big Data approach.

In this paper, we focus on the use of Big Data paradigm in industrial automation, including manufacturing, transportation, energy consumption etc. We discuss the

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 305–316, 2013.

applications and connected challenges. We review possible further steps and opportunities. We pay special attention to the semantics and integration of data. Also, the Big Data are inherently of distributed nature and that's why we also discuss possibilities of the multi-agent systems (MAS) paradigm use especially in the industrial use cases.

The paper is organized as follows: First we discuss the term Big Data, elaborate its properties and show how this paradigm is different from just large databases. Then we elaborate some of the possible Big Data applications that are related to industrial automation and manufacturing. We summarize the common features and challenges. Then we focus on two selected dominant synergies – Big Data together with semantics and multi-agent systems. We conclude with summarizing next steps for research.

2 Big Data

The Big Data concept means a dataset that is growing so that it becomes difficult to manage it using existing database management concepts and tools [18]. The difficulties may be related to data capture, storage, search, sharing, analytics etc.

The main reason for the necessity of using these data is better data driven decision making. Ideally, predictive intelligence should be available based on the data – that includes data connectivity and prediction together with making actions towards beneficial events and avoiding adversary events – than just reactive intelligence for solving problems that already occurred. The ability to handle these data presents an opportunity to create unprecedented business advantage; however, it also requires new infrastructures and new ways of thinking different from the classical business intelligence approaches.

The term Big Data is usually characterized by the following dimensions – Volume, Velocity and Variety (also known as three Vs):

- Volume – the amount of data is typically large, in the order of terabytes or larger
- Velocity – the access to data using appropriate queries is needed in "real" time
- Variety – the data are unstructured, possibly including texts, videos, log files etc.

2.1 Volume

The IDC (International Data Corporation) study predicts that the overall data will grow by 50 times by 2020. The contributions will by a significant part from the more embedded systems such as sensors in clothing, medical devices and structures like buildings and bridges [18]. As we have already mentioned, producing terabytes of data from a plant already happens today. As discussed in [9], the problem is both in the acquisition of the data as well as in storing them to be accessible. The RDBMS (Relational Database Management Systems) can scale to this amount of data – if the data are well structured with the schema known ahead – at least for the storage.

However, for some uses, the NoSQL databases are more appropriate – some of the RDBMS properties (typically the ACID properties – atomicity, consistency, isolation, durability) are dropped so that the database can exhibit other features, such as flexible schema and elasticity, fault tolerance and massive scalability in processing the data. Some of the possibilities of NoSQL databases include key-value stores (Membase), column-oriented stores (Google BigTable, HBase), document stores (CouchDB, MongoDB) and graph database systems (Neo4j) [4].

For processing the data, MapReduce framework [7] is usually employed. In this framework, every data processing task is divided into two steps: the map step divides the task into independent subtasks that are executed in parallel, then reduce steps recursively (possibly also in parallel) merge the results from subtasks. The technology to handle large amount of data is in rapid development, however, from the overview above we can see that for this dimension suitable solutions are already being offered.

2.2 Velocity

We have already touched the velocity as it is tightly connected to the volume. The speed of both storing data (whatever data are available) and getting stored data (either from simple key-values stores or from complicated queries) is very important. An example of time sensitive process is catching a fraud – the decision must be made within at most a few seconds.

For both storing and retrieving the data, farms of many data store machines are typically used, so that these operations can be balanced between them as needed. If the queries can be formulated in a way that they can run on clusters of data with aggregating the results together, then the MapReduce approach [7] can be used to achieve massive parallelism with a great scalability.

2.3 Variety

The other characteristic of Big Data is that much of the data is unstructured. This is in contrast to well organized relational databases. It is true that even for relational databases there are ways for solving the integration of different schemas. However, with Big Data, the degree of difference between data is much higher – consider for example integrating streams of data from sensors with videos from multiple cameras and text documents in a meaningful way.

The IDC study mentioned above also predicts that unstructured information – such as files, e-mails, videos – will account for 90% of all data created over the next decade [18]. One of the main outcomes of the survey [15] among the Fortune 500 leading companies is that "it's about variety, not volume" and that data integration will continue to be one of the greatest challenges faced. Another relevant characteristic from [17] is that "Big Data is a Big Mess". The Variety dimension is truly one of the most important challenges to be solved. For the more automated integration of data, adding proper semantics is needed, for example using annotations.

2.4 Other Factors

The three V factors described above are used as the main ones characterizing the Big Data concept. As we have seen, some of them may be more prevalent than others depending on particular application. However, there are also dimensions that can be at least equally important – we discuss some of them in this section.

Veracity. The veracity dimension, also known as "data in doubt", deals with uncertain or imprecise data. In classical business intelligence analyses it is believed that this is not an issue, since it is possible to cleanse the data before using them. However, with the Big Data velocity and variety, the veracity, including the trust in the information received, is most often cited as the additional important fourth V dimension. The issues to be dealt with include data inconsistency, incompleteness, ambiguities, latency, deception and approximations.

Validity. Another related dimension is validity – when operating on big wrong data the results would be unusable or misleading. The validity dimension includes not only ensuring that the measurements are correct, but also transparency of the assumptions and connections behind the process. Again, the variety of data play important role – achieving validity often means making the assumptions explicit and justifiable. The validity can be further divided into content validity, structural fidelity, criterion validity and consequential validity.

Security. Security is involved in various phases of the work with Big Data, including gathering data, analyzing them, visualizing them and using them for actions. In all of these phases it must be ensured that data are accessed only by agents with appropriate rights. It is still difficult to understand how exploiting variety of data with different access rights can affect the results of work with Big Data.

Privacy. Privacy is tightly connected with security dimension. Inappropriate use of personal data, such as through linking of data from multiple sources, including location-based services, is becoming one of the central problems in using of web data for online marketing. This concern is even more visible when working with sensitive information such as electronic health records [5] or with manufacturing/production data.

2.5 Examples of Applications

For illustration, let us very briefly mention a few often cited examples of Big Data applications.

Online Marketing. Web based companies such as Google, Amazon, Twitter, Netflix use Big Data processing for targeting products and ads for their user based on the information collected about users, particularly about the user clickstreams.

Medicine, Biology, Chemistry. The research in these areas includes more and more analysis of available data collected from various sources. The classical data pumps and data mining approaches are becoming insufficient for handling Big Data with the properties described above.

Floods and other Disasters. An example of Big Data with all the Vs and other properties as described above are data from disasters. During such events, data are coming with a great velocity and decisions and actions are needed with the same speed. The data mashups and parallel processing at various levels are obvious in this case.

2.6 Difference to Traditional Business Intelligence Approaches

Let us briefly compare the Big Data approach to the business intelligence (BI) approach. As we already mentioned, Big Data is more than only large scale data – more properties of data are involved that are not traditionally present or reflected in BI.

The classical BI approach is as follows. Data are gathered from an OLTP (Online Transaction Processing) database that is optimized for fast multiuser use. Data are then on request stored into an OLAP (Online Analytical Processing) data warehouse that is organized in a way for efficient answering of queries for summaries and trends – so called data cubes with predefined dimensions are often used. The use for reporting, dashboarding and predictions including data mining is then possible – as far as the predefined dimensions within the cleaned data allow.

On the other hand, Big Data processing is more targeted to information discovery in the conditions described above. The operation on data is using massive parallelism, often using map-reduce paradigm, so that demanding one purpose data preprocessing is not present. The velocity of Big Data makes such preprocessing impossible – online processing is needed instead of offline analysis. The BI top-down approach is replaced by bottom-up approach that uses mashups and visualizations instead of predefined reports. The process is often driven by business questions and the main use case is interactive analysis (ask and see) with immediate reactions instead of monitoring and reporting that is developed and refined in advance.

2.7 Main Big Data Factors

Let us emphasize the main points that we believe are important to realize for the Big Data applications (some of them were mentioned above):

- The improved data driven decision making is the main driving force [18, 15],
- "It is about variety, not volume" [15, 17] – heterogeneous data integration is much more important than the volume of data,
- A better term for many applications is "mashup data" instead of "Big Data" [15].

As we can see, the focus is not on the "big" part of the Big Data, but on the problem of data variety, heterogeneity and the use for decision making in appropriate time.

3 Applications Related to Industrial Automation and Manufacturing

It is clear that the industrial automation and manufacturing sector is already a large producer and consumer of a lot of data. During the last few decades, manufacturing companies demonstrate significant productivity gains, both in terms of improved quality and efficiency, based on utilization of process data and using advanced methods for data analysis.

Today, the Big Data paradigm is a promising concept for another substantial wave of gains to achieve improved efficiency in design, production, product quality and following customers' needs. According to [14], Big Data can help manufacturers to reduce product development time by 20 to 50 percent and to eliminate defects prior to production through simulation and testing. Using real-time data, companies can also manage demand planning across extended enterprises and global supply chains while reducing defects and rework within production plants.

Table 1. Big Data levers across the manufacturing value chain [14]

	R&D and design	Supply-chain	Production	Marketing, sales	After sales
Interoperable R&D and product databases	X				
Aggregation of customer data	X			X	
Virtual collaboration sites	X			X	
Demand forecasting		X	X	X	
Lean manufacturing and production model			X		
Sensor data-driven operations analytics			X		
After sales data from sensors			X	X	X

3.1 Big Data Levers

There are several potential performance improvement areas for manufacturers identified by McKinsey Global institute for Big Data utilization, see the Table 1. The table spans from R&D through supply chain and production to marketing and after sales services and as we can see, various aspects where Big Data can help play important roles. As we can see, the range of possibilities is quite wide. We focus mostly on the manufacturing and data driven operations in the rest of this paper, however, the combination of the data from sales and marketing (as we know it from internet applications) together with the data stored and processed during production will lead to much more efficient production, including quality and costs savings as well.

3.2 Energy and Water Consumption and Production

The electrical energy consumption is of distributed nature for a long time; however, the production has become more distributed as well. In contrary to the classical centralized power plant model, smart grid consists of small generators which are scattered geographically and involves many uncertain facts as well – intermittency of wind and solar generation, electrical vehicles and charging stations that include both charge and discharge from the grid. This mix of resources requires more real-time and more granular forecasting, load planning and commitment analysis. Today, many utilities are moving to smart meters and grids as part of long-range plans to ensure a reliable energy supply, to incorporate distributed generation resources, to develop innovative storage solution, to reduce the need to build new power plants and to enable customers to have more control over their energy use [10]. Hence, organizations involved in energy consumption and production have to deal with Big Data problem because of the volume of information from smart meters and smart grids changing environment. With the usage of predictive analytics on the data, there is a wide range of forecasts, including the amount of available electricity, the best time to sell or buy the energy, transmitting capacity of the grid, time and reasons of possible failures, etc. Most of these forecasts are needed in real-time (velocity), the amount of data to be processed is huge (volume) and the data come from disparate sources (variety) – producers, consumers, grid (load, failures, and contracts), weather (affecting both production and consumption), etc. Similarly as with electrical energy consumption we clearly see the fit of the Big Data paradigm for any other similar energy or commodity distribution – including distribution of water, gas, petrol, etc.

3.3 Transportation

The capacity of many transportation networks (for example railways and roads, especially in large cities) is at least sometimes overloaded. A minor accident can transform normally pervious network in a chaos with unwanted delays for thousands of travelers and transport companies. A transport network is commonly composed of numerous service companies, each with their own systems, data, and targets. Consequently, data collection, analysis and dissemination remain fragmented and unreliable and so it is hard to communicate the information to the travelling public [8]. The solution to overcome these drawbacks is to integrate transport data from the involved subjects and turn collected data into valuable information for all involved parties (i.e., for travelers and transport companies). Of course, all of this information has to be also connected to an actual state of the transportation network and its current load. Ideally, transportation system can be intelligent and interconnected, avoiding wasting time in traffic jams and saving total transportation costs.

To achieve this goal, the Big Data tools and technology may help. We again see the need for handling volume, variety and velocity of data (in fact, the transportation as described is in some aspects not far from energy or water distribution). In addition, the distribution of proper information to transport entities may be also a challenge.

For manufacturing companies, the intelligent supply chains are also affected by the available transportation networks. Especially for companies using a truly lean manufacturing with just-in-time (JIT) transportation any disruption in transportation, including traffic jams, may have great impact on their production. It was also reported that in a small area the JIT approach itself led to traffic jams on local roads.

Handling Big Data of disparate origin that contribute to understanding and possibly predicting the transportation situation would help avoiding these problems at least partially even when the transportation network capacity is becoming unsuitable.

3.4 Internet of Things

We have already mentioned the need of smart meters and distributed sensors above for efficient decision making in some of the specialized applications. However, the idea can be taken to a higher level when these devices are communicating over the internet – they are embedded in the physical world and connected by networks to computing resources forming so called Internet of Things (IoT). Even today, the smart objects like mobile phones form a big part of internet communication. This trend is another one that is driving the growth in Big Data [6]. The forecast for the amount of connected nodes in the IoT predicts 30 percent growth annually over the next five years. Some of the growth sectors are expected to be health care (deploy remote health monitoring), retail (RFID tags), and automotive industry (sensors in vehicles). The next consequence of the proliferation of IoT can be the optimization of operations by embedding real-time data from networked sensors in supply chains and production processes. Data fusion and data integration in IoT domain are assured for example through utilization of signal processing techniques. According to [2], the use of Big Data technologies within IoT will enable forming smart homes, cities, and vehicles and will bring many advantages various industry sectors.

3.5 Plant Data Processing

Manufacturing plants collect time series and other data for various purposes. These data are kept to have access to historical information when needed, for example for diagnostics, various audits, safety enforcements, etc. The historian software and hardware that is usually provided is typically focused on the ability of collecting and storing the data in suitable scan rates. Usually, it is possible to collect real time process and production data at the machine levels and then to store them to a higher level data collections, such as at the level of a manufacturing site. Sometimes, the basic analysis at the levels of machines is possible – for example detecting trends or outliers, selecting only interesting or important values to be stored, etc.

As discussed in [9], current historians are able to handle the storing phase of data, even for larger amounts of data. Simple retrieving of the stored data is usually not a problem as well – especially when the data stored are not of too big volume, if data are requested based on time and if data are asked in time sequence order.

However, any further data processing and more advance queries are beyond the typical historian software. For example, looking for a pattern in data similar to a given one is something that can be easily solved by a map-reduce approach – the query is distributed to the nodes holding the data, the query is processed in parallel, and then the results are collected. The pattern may represent something that has happened before a machine failure or before producing a set of products of a low quality. The query does not have to be a simple retrieval, additional processing can occur as well – for example, we may want to ask for an average temperature measured on certain devices an hour before an alarm occurred. All of these queries are almost impossible to be asked using currently available historians, especially within large volume of data. When Big Data approach is used for storing and processing the data, these types of queries can be evaluated allowing more ad-hoc analysis. On the other hand, the access to small rapidly changing data will be still better solved by existing historians.

Another important feature that Big Data may allow is introducing higher variety of data. The data in historians need to be annotated – it is necessary to know from which machine and place the data came from. However, this can be augmented by other available data, such as shift change, indications and causes of outages, weather conditions (for operations where conditions such as air pressure and temperature may play important role) or even the plant operation recorded on cameras. Integrating these data together is something that can make the analysis and decision making more accurate, finding patterns and causes that would remain hidden otherwise.

3.6 Summary of Features and Challenges

From the applications briefly described above the following common and dominant specific features and challenges can be emphasized:

- Any distribution and/or transportation presents a good fit for Big Data paradigm – the connectivity from and to heterogeneous, spatially distributed and possibly moving and not always connected data sources presents one of the challenges.
- In many areas, data are available, but are not processed since there is no technology to really exploit such data; however, at the same time it must be kept in mind that relevant data must be collected (logs that were just produced and deleted so far might be typically of little use) and made available for processing while reflecting security and privacy constraints at the same time.
- The challenge is integrating and using data from disparate sources – addressing using web technologies (URIs, IPv6) and tagging using RFID is expected in manufacturing applications. Semantic Web technologies may bring the semantics.
- Available Big Data technologies allow processing ad hoc queries even within large volumes of data, i.e., something that was not possible until now. However, the better decision making would be useless without actions – and connecting the decisions to actions is a challenge especially in the physical world where a wrong action of industrial automation system may have more serious consequences when compared to for example wrong recommendation by internet marketing companies.

4 Related Challenges

Let us elaborate some of the challenges mentioned above in the relation to industrial automation applications, especially heterogeneity of the data (the need of semantics) as well as inherently parallel Big Data processing where the multi-agent paradigm can be employed. Other challenges include: search (for example within plant data), exploration (i.e., exploring without knowing exact query), visualization (including spatially segmented visualization) and decision making transformed to actions. Of course, the Big Date factors described earlier are present in these applications as well.

4.1 Semantics

As we have mentioned earlier, the variety of data is often overlooked at the first sight, but becomes the one most difficult and important for many applications – unlike the volume of data, for which appropriate technologies already exist, at least as a base for further development.

Semantic Web. The problem of the variety of data is being solved within the Semantic Web. Semantic web is an activity to enrich documents available on WWW so that they would have meaning understandable to computers. The problems of resolving, identifying and combining entities is solved by URIs (Uniform Resource Identifier), the problem of storing and connecting structured data is solved by using Resource Description Framework (RDF) [13], the problem of adding semantics for integration and further processing is being solved by using ontologies and knowledge bases described in Web Ontology Language (OWL) [19]. Sound and complete reasoning over data is possible as well.

Linked Data. The principles defined within Linked Data can be utilized [3]. Various proprietary vocabularies are used to describe structured data stored in RDF or JSON (JavaScript Object Notation). It is possible to relate these data together using identity and vocabulary links as provided by RDF. This allows data fusion in Big Data warehouses for further processing. However, the definition of these links will be still at least partially a task that may not be fully automated – obtaining the mappings is still issue in Linked Data. Let us also note that some of the data may be schemaless and that maintaining the links in NoSQL databases is also an active research topic.

Structured Data Processing. The time series data processing typically does not require complex semantics, except annotating the source and time of data. However, other more structured and complex data mentioned in the applications above require more attention even in the phase of storing the data. The NoSQL databases that work with documents allow storing JSON and RDF data, however, the proper linking and cleanup of such data needs to be made to allow efficient run of map-reduce tasks.

Limits. As we can see, the Semantic Web (and particularly Linked Data) principles and approaches provide ways for a meaningful use of Big Data with high level of variety. The fact that current Linked Data available on the web exceed tens of billions of RDF triples [3] also indicates the amount of data processable using current tools.

4.2 Multi-Agent Systems

Big Data processing represents a complex task which in addition to its parallelism is highly dynamic. The MAS (Multi-Agent System) paradigm seems to be a suitable solution for at least some of the problems of the Big Data analysis when compared to the classical centralized solution. The features such as distribution and parallelism are present in the use of Big Data by internet companies, but these factors are different in the applications described in this paper, mainly because of acting directly in the physical world. Besides parallel programming tasks, the MAS approach can be a very suitable complement to the Big Data processing, including existing agent-based frameworks for high performance computing, such as REPAST [16].

Spatial Distribution. The spatial nature of Big Data corresponds with a nature of MAS. Agents are able to act autonomously, to carry out the satisfaction of their own goals, and are able to interact with other agents. The spatially distributed agents can handle both gathering the appropriate data and providing the actions based on decisions made. As we have seen above, some of the applications are inherently spatially distributed, including smart grids and transportation networks.

Heterogeneity of Data Sources. Usually, Big Data tasks include many sources with different data format which presents important obstacle during data processing. In MAS, agents are heterogeneous in their behavior that varies in sophistication, granularity and information load for decision making. As a result, the agent-based paradigm is very well-suited to representation of complex heterogeneous systems [11]. This is also connected to the semantics and data variety discussed above. The data acquisition from heterogeneous sources is something that agents are suitable for, and again, the same is true for performing required actions in various environments.

Real-Time Processing. Big Data analysis has to be processed in real-time or near real-time (e.g., face detection – airport security, earthquakes detection in tsunami-prone Japan). Benefits of using MAS technology for real-time processing include:

- Increased speed and efficiency due to parallel and asynchronous computation.
- Increased robustness – the systems degrade gracefully when an agent fails.
- Increased scalability – agents can be added as and when necessary [1].

5 Summary, Next Steps and Conclusions

We have outlined how the Big Data paradigm is already helpful for the industrial automation and manufacturing and how this approach can help with many currently unsolved problems. We have also described some of the possible applications and challenges. As we can see, even when the Big Data approaches were developed by internet companies, the methods can be applied in the industrial automation domain as well. This is similar to the Semantic Web – even when originally developed for the WWW, some of its technologies can be applied for industrial automation as well, especially when operating on levels above a single controller. Working at the level of plants, cities, virtual enterprises etc., certainly requires handling Big Data.

We have also briefly elaborated two areas that we see as important for our further focus – bringing the semantics to cope with Big Data variety and multi-agent systems paradigm to cope with the massive parallelism. The value of Big Data usage is better decision making – and performing the actions based on the decisions is another area where the multi-agent paradigm is a good complement to Big Data technologies.

5.1 Next Steps

To summarize, the important challenges to be elaborated and solved include:

- Integrating data from various sources – e.g., from multiple places in a plant, in a transportation network or a smart grid – while resolving their heterogeneity,
- Infrastructure to provide means for efficient decision making – this includes search, visualization, exploration, and means for analysis dependent on application,
- Provide means for distributing actions based on decisions made from Big Data analysis – this is an important counterpart to data collection and integration.

5.2 Conclusion

From the overview we can conclude that Big Data technologies applications within the industrial automation domain will not only enable more efficient decision handling, but also that in fact they are necessary for further improvements within this domain. The important next steps for enabling the use of Big Data in the industrial automation related applications were outlined in this paper.

References

1. Balaji, P.G., Srinivasan, D.: Multi-Agent System in Urban Traffic Signal Control. IEEE Computational Intelligence Magazine 5(4), 43–51 (2010)
2. Becker, A., Sénéclauye, G., Purswani, P., Karekar, S.: Internet of Things. Atos White Paper (2012)
3. Bizer, C., Boncz, P., Brodie, M.L., Erling, O.: The Meaningful Use of Big Data: Four Perspectives – Four Challenges. SIGMOD Records 40(4) (December 2011)
4. Carme, J., Jimenez, F.J.R.: Open Source Solutions for Big Data Management. Atos White Paper (2011)
5. Community white paper: Challenges and Opportunities with Big Data (2012)
6. Chui, M., Löffler, M., Roberts, R.: The Internet of Things. McKinsey Quarterly (2010)
7. Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM 51(1), 107–113 (2008)
8. Detica: Unblocking the transport network. Whitepaper (2010)
9. GE Intelligent Platforms: The Rise of Industrial Big Data. Whitepaper (2012)
10. IBM Software: Managing Big Data for smart grids and smart meters. Whitepaper (2012)
11. Lin, J., Sedigh, S., Hurson, A.R.: An Agent-Based Approach to Reconciling Data Heterogeneity in Cyber-Physical Systems. In: IEEE International Symposium on Parallel and Distributed Processing, pp. 93–103 (2011)
12. Kaisler, S., Armour, F., Espinosa, J.A., Money, W.: Big Data: Issues and Challenges Moving Forward. In: 46th Hawaii International Conference on System Sciences. IEEE Press (2013)
13. Manola, F., Miller, E. (eds.): RDF Primer. W3C Recommendation (2004)
14. Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C.: Big Data: The next frontier for innovation, competition, and productivity. McKinsey Global Institute Report (2011)
15. NewVantage Partners: Big Data Executive Survey 2012. Consolidated Summary Report (2012)
16. North, M.J., Collier, N.T., Ozik, J., Tatara, E., Altaweel, M., Macal, C.M., Bragen, M., Sydelko, P.: Complex Adaptive Systems Modeling with Repast Simphony. In: Complex Adaptive Systems Modeling. Springer (2013)
17. Schlieski, T., Johnson, B.D.: Entertainment in the Age of Big Data. Proceedings of the IEEE 100, 1404–1408 (2012)
18. Singh, S., Singh, N.: Big Data analytics. In: 2012 International Conference on Communication, Information & Computing Technology (ICCICT), Mumbai, India. IEEE Press (2012)
19. W3C OWL Working Group. OWL 2 Web Ontology Language Document Overview, 2nd edn. W3C Recommendation (2012)

Author Index