TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

Mohammad Kamal Uddin

An Application of Context-sensitive Computing for Flexible Manufacturing System Optimization

Julkaisu 1468 • Publication 1468

Tampere 2017

Tampereen teknillinen yliopisto. Julkaisu 1468
Tampere University of Technology. Publication 1468

Mohammad Kamal Uddin

An Application of Context-sensitive Computing for Flexible Manufacturing System Optimization

Thesis for the degree of Doctor of Science in Technology to be presented with due permission for public examination and criticism in Festia Building, Auditorium Pieni Sali 1, at Tampere University of Technology, on the 26th of May 2017, at 12 noon.

Tampereen teknillinen yliopisto - Tampere University of Technology
Tampere 2017

ISBN 978-952-15-3934-3 (printed)
ISBN 978-952-15-3956-5 (PDF)
ISSN 1459-2045

Tampere University of Technology

Mohammad Kamal Uddin

An Application of Context-sensitive computing for Flexible Manufacturing System Optimization

Tampereen Teknillinen Yliopisto

Tampere University of Technology

Uddin, Mohammad Kamal: An Application of Context-sensitive computing for Flexible Manufacturing System Optimization

Tampere University of Technology, Faculty of Engineering Sciences, Finland 2017

Keywords: Context, Flexible manufacturing system (FMS), Optimization, Key performance indicator (KPI), Service oriented architecture (SOA), Web Service (WS), Web Ontology Language (OWL).

Abstract

Recent advancements in embedded systems, computing, networking, WS and SOA have opened the door for seamless integration of plant floor devices to higher enterprise level applications. Semantic web technologies, knowledge-based systems, context-sensitive computing and associated application development are widely explored in this regard. Ubiquitous and pervasive computing are the main domains of interest among many researchers so far. However, context-sensitive computing in manufacturing, particularly, relevant research and development in a production environment like FMS is relatively new and growing.

Dynamic job (re)scheduling and dispatching are becoming an essential part of modern FMS controls. The foremost drive is to deal with the chaotic nature of the production environment while keeping plant performance indicators unaffected. Process plans in FMS need to consider several dynamic factors, like demand fluctuations, extreme product customizations and run time priority changes. To meet this plant level dynamism, complex control architectures are used to provide an automatic response to the unexpected events. These runtime responses deal with final moment change of the control parameters that eventually influences the key performance indicators (KPIs) like machine utilization rate and overall equipment effectiveness (OEE). In response, plant controls are moving towards more decentralized and adaptive architectures, promoting integration of different support applications. The applications aim to optimize the plant operations in terms of autonomous decision making, adaptation to sudden failure, system (re) configuration and response to unexpected events for global factory optimization.

The research work documented in this thesis presents the advantages of bridging the mentioned two domains of context-sensitive computing and FMS optimization, mainly

to facilitate context management at factory floor for improved transparency and to better respond for real time optimization through context-based optimization support system.

This manuscript presents a context-sensitive optimization approach for FMS, considering machine utilization rate and overall equipment effectiveness (OEE) as the KPIs. Runtime contextual entities are used to monitor KPIs continuously to update an ontology-based context model, and subsequently convert it into business relevant information via context management. The delivered high level knowledge is further utilized by an optimization support system (OSS) to infer: optimal job (re) scheduling and dispatching, keeping a higher machine utilization rate at runtime. The proposed solution is presented as add-on functionality for FMS control, where a modular development of the overall approach provides the solution generic and extendable across other domains. The key components are functionally implemented to a practical FMS use-case within SOA and WS-based control architecture, resulting improvement of the machine utilization rate and the enhancement of the OEE at runtime.

Acknowledgements

This has been quite an exciting and long journey with many landmarks, challenges and success stories in it. Many actors and their positive interactions in different stages helped the journey to find its destination. It would be cumbersome to mention all those names here, however, I would like to take this opportunity to mention at least some names who have been influential in getting this thesis to completion.

***Pre-examiners, Prof. Haber** and **Prof. Harrison** Thank you for reviewing the thesis. It's an honour for me.*

***Prof. Anderson**: Thank you for your co-operation and confidence in me during my entry to the department of production engineering in 2007. Rest in peace.*

***Sonja, Taina and Hanna**: Thank you for always being there for me with your lovely smiles during many of those travel arrangements and other practical matters.*

***Niklas and Tomi**: My Wärtsilä colleagues, Thank you for your motivational words from time to time.*

***Andrei, Corina, Jani and Alexandra**: Thank you for your advices, fruitful discussions at work, especially during the writing phase of the research plan and publications.*

***Juha**: Thank you for your programming support and many good discussions we had during the project works.*

***Zahid, Aziz, Miraj, Rifat and Shisir**: Thank you for all the motivation, encouragements and for always being there for me and for my family.*

***Yousuf Ali**: My father in law, Thank you for your inspiration and for reminding me your desire about my doctoral degree from time to time.*

***Nasrin and Yeasmin**: My lovely elder sisters, your love and affection helped and still help me to grow.*

***Ema, Sadid and Samanta**: My beautiful wife, my son and my daughter, without your presence, love and care things would've been impossible. Thanks Ema for your true love and enormous patience. I love you.*

***Khabir Uddin and Rehana Khatun:** My beloved parents, your unconditional love, big sacrifices and supports have brought me here. You've taught me how to dream big and how to achieve it. I dedicate this work to you.*

*And last but not least, **Prof. Lastra**: It's an honour and privilege to work under your supervision. Thanks for giving me the opportunity to explore myself in FAST lab under your guidance and leadership. Again, my heartfelt thanks to you for everything.*

Kamal Uddin
Vaasa, Finland
05 April, 2017.

Foreword

The research outcome reported in this thesis was carried out within the Department of Production Engineering in Tampere University of Technology, Finland, during the period 2008-2013. Financial support for this work has been provided through:

- The Self-Learning (Reliable Self-Learning Production Systems based on Context Aware Services) project of European Union's 7th Framework Program, under the grant agreement no. NMP-2008-228857.
- Grant from Wärtsilä Foundation of Tampere University of Technology, Finland.

“After climbing a great hill, one only finds that there are many more hills to climb”
- Nelson Mandela

সাদিদ এবং সামান্তাকে:
“ বাবা যখন সায়াহ্নে, এ জীবন গড়তে হবে
বিপদ, ভীতি, দুঃখ, ব্যাথা - শক্ত হাতে লড়তে হবে ”

Contents

List of Figures

List of Tables

List of Acronyms

AmI	Ambient Intelligence
AL	Assembly Line
API	Application Programming Interface
CORBA	Common Object Request Broker Architecture
DCE	Distributed Computing Environment
DCOM	Distributed Component Object Model
DFM	Design for Manufacturing
DPWS	Device Profile for Web Services
ECA	Event Condition Action
ERP	Enterprise Resource Planning
FMS	Flexible Manufacturing System
GUI	Graphical User Interface
IoT	Internet of Things
J2EE	Java 2 Platform, Enterprise Edition
KPI	Key Performance Indicator
KR	Knowledge Representation
MES	Manufacturing Execution System
MMALs	Mixed-model assembly lines
NC	Numerically Controlled
OEE	Overall Equipment Effectiveness

OOM	Object Oriented Model
OSS	Optimization Support System
OWL	Web Ontology Language
PLC	Programmable Logic Controller
RDF	Resource Description Framework
RDFS	RDF Schema
RFID	Radio Frequency Identification
SCADA	Supervisory Control and Data Acquisition
SDB	Storage and Query Database of RDF data
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
SWRL	Semantic Web Rule Language
UI	User Interface
UML	Unified Modelling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WS	Web Services
WSDL	Web Service Description Language
XML	Extensible Markup Language

Refereed Publications

This dissertation consists of an overview and the following peer-reviewed publications, which are referred to in the text by their Roman numerals.

I. Uddin, M. K., Soto, M. C., Martinez Lastra, J. L., "An integrated approach to mixed-model assembly line balancing and sequencing", *Assembly Automation*, vol. 30(2) pp.164 – 172, 2010.

II. Uddin, M. K., Dvoryanchikova, A., Lobov, A., Martinez Lastra, J.L., "An ontology-based semantic foundation for flexible manufacturing systems," *37th Annual Conference on IEEE Industrial Electronics Society, IECON*, vol., no., pp.340,345, 7-10, 2011.

III. Uddin, M. K., Dvoryanchikova, A., Martinez Lastra, J.L., Scholze, S., Stokic, D., Candido, G., Barata, J., "Service oriented computing to Self-Learning production system," *9th IEEE International Conference on Industrial Informatics (INDIN)*, vol., no., pp.212,217, 26-29, 2011.

IV. Uddin, M. K., Puttonen, J., Scholze, S., Dvoryanchikova, A., Martinez Lastra, J. L., "Ontology-based context-sensitive computing for FMS optimization", A*ssembly Automation*, vol. 32(2), pp.163 – 174, 2012.

V. Uddin, M. K., Puttonen, J., Martinez Lastra, J. L., "Context-sensitive optimization of the key performance indicators for FMS", *International Journal of Computer Integrated Manufacturing*, vol. 28(9), pp. 958-971, 2014.

Author's Contribution

Publication I "An integrated approach to mixed-model assembly line balancing and sequencing"

The majority of the work in this article was contributed by the doctoral student, including the concept and the development of the reported methodology and implementation. Prof. Lastra and Prof. Soto contributed to the writing style and review the visual representation of the article.

Publication II "An ontology-based semantic foundation for flexible manufacturing systems"

The background and contributions of this paper was generated by the doctoral student following his work at the European project 'Self- Learning'. Dr. Dvoryanchikova, Dr. Lobov and Prof. Lastra contributed to the writing style and provided support during the conference oral presentation.

Publication III "Service oriented computing to Self-Learning production system"

Most of the work reported in this publication was contributed by the doctoral student. Dr. Dvoryanchikova and Prof. Lastra participated during the writing process. The presence of the other author's name is justified by the general guideline outlined by the European project 'Self-learning' for wider dissemination of the project outcomes. The other authors are the project partners within the European project 'Self- Learning'.

Publication IV "Ontology-based context-sensitive computing for FMS optimization"

The scientific work was done by the doctoral student within the research scope of the European project 'Self-Learning'. The work was supervised by the European project co-ordinator, Dr. Scholze and by the close supervisor at TUT, Prof Lastra. Dr. Dvoryanchikova provided support for the writing style and Dr. Puttonen provided the programming work for the interfaces.

Publication V "Context-sensitive optimization of the key performance indicators for FMS"

The work reported in this manuscript was contributed by the doctoral student, supervised by Prof. Lastra. In order to provide a working prototype for the FMS use case, Dr. Puttonen provided programming support for interface and GUI development.

1 Introduction

This chapter provides an introduction to the research work presented in this dissertation. It covers the background, hypothesis, objectives and the aimed contributions of this thesis.

1.1 Background

The ever challenging nature of the global economy and trade has resulted in high competition that has led manufacturers to face a vibrant operating environment. Such an environment has to deal with, for instance, volume uncertainty, rapid market changes, increased product variety, competitive prices, on time delivery and short product life cycles. The modern manufacturing paradigms, are therefore moving towards more flexible systems and operations, coupled with intelligent and adaptive control paradigms, so that these uncertainties can be handled effectively without compromising the performance indicators.

The concept of ‘anytime and anywhere’ is being introduced by pervasive computing that is becoming increasingly present in our daily tasks through a wide variety of smart devices (e.g. mobile phones). The recent focus is to bring this emerging paradigm at the plant floor level to enable a comprehensive domain knowledge and utilization of this knowledge for users and for support applications via a standardized interface. Knowledge and knowledge management, context-sensitive computing is the part of this dynamic scenario. Generic and dynamically updated, managed context models are of interest in this regard since such a model is reusable and enables contextual knowledge sharing between systems.

However, application of context awareness, especially for optimization research in a dynamically changing operating environment of FMS is still in an early phase.

1.2 Problem Description

Conventional manufacturing at factory level is known to have a number of limitations, as different manufacturing states are isolated and cannot provide the necessary transparency since there is a lack of infrastructure providing holistic and explicit domain knowledge. This lack of insight prevents optimal decision making in real-time.

Manufacturing systems design in FMS faces many challenges due to the varying and evolving nature of the environment as demand change, customization of products, production priorities

instability, keeping the due delivery date. This often requires a final moment change in the control parameters which in turn poses significant challenges to the global factory optimization by affecting the KPIs.

1.2.1 Problem Statement

The problem statement can be framed as follows:

> *"How can context-sensitive optimization be addressed in an SOA-based dynamic operating environment of FMS for run time KPI optimization?"*

The publications included in this manuscript answers the above problem statement as follows:

- *How to address optimization identifying the KPIs?* This question is answered in Publication I, which provides a basis for optimization technique using the KPIs in a chaotic operating environment such as mixed-model assembly lines.
- *How manufacturing semantics can be utilized in FMS?* Publication II answers this question by bridging ontology and lower plant level data as the foundation for building a context-based decision support system.
- *What are the advantages of SOA, as an architectural paradigm for emerging production technologies?* Publication III answers this question with a focus on bridging of SOA with modern production technologies such as Self-learning production system.
- *How manufacturing context, captured from SOA platform can be utilized for runtime KPI optimization in FMS?* Publication IV and V answer this question by providing a novel methodology for utilizing context-sensitive computing for runtime FMS optimization.

1.3 Research Description

1.3.1 Hypothesis

The main hypothesis of this research work is that, continuous improvement of the factory can be enhanced significantly utilizing knowledge-based context models which provides intelligent interface for knowledge acquisition and elicitation. Further use of this model enables improved data analysis and diagnostics, feedback control dynamically and provide optimization support.

1.3.2 Objectives

The main objective of this thesis is to apply a knowledge-based approach for context-sensitive optimization to achieve run time optimal job re-scheduling and dispatching in FMS, which in turn provides continuous improvement of the KPIs.

1.3.3 Contributions

This thesis presents the following original contributions:

- A new approach for runtime optimization methodology at the KPI level in a dynamic operating environment of FMS based on context-sensitive computing.

- The above methodology provides a novel architecture for process, resource and product level context extraction from an SOA-based platform and updates and manages those for higher level processing through an ontology-based context model.

- The developed ontology-based context model for FMS also allows domain specific extensibility and a modular development of the overall approach makes the solution generic and extendable across other domains.

- A new context-based optimization support system and the underlying algorithm, which consumes and adapts KPI relevant contents from periodically updated knowledge contexts and proposes an optimal job dispatching order in a GUI, enabling decision support for global factory optimization.

1.3.4 Limitations of Scope

This thesis demonstrates the applicability of the proposed context-sensitive approach in FMS environment. The focus is on optimization and context-sensitive computing used to solve optimization problems. Comparison of the proposed approach to other modelling and optimization techniques is considered out of scope of the work carried out in this research.

1.4 Thesis Outline

Chapter 1 introduces the topic and the main contributions of this work. In addition, it presents the problem definition, research hypothesis, objectives and limitations of scope. In Chapter 2, prior work in this field is reviewed presenting the current state of the art.

Chapter 3 presents the concise results of all the publications included in this thesis and finally

it draws a conclusion from the publications.

Chapter 4 summarizes the contributions, lessons learned and outlines future research directions within this area. Table 1 presents the thesis structure as follows:

Table 1. Thesis structure.

Chapter 1	Introduction
Chapter 2	Literature and technology review
Chapter 3	Context-sensitive optimization for FMS
Chapter 4	Conclusions and recommendation for future works

2 Literature and Technology Review

This chapter presents a review and assessment of prior work carried out by others.

To begin with, the review focuses on the notion of context and context-sensitive computing. Identifying different context modelling approaches and underlying core requirements for context modelling, ontology-based development of context-sensitive computing is analysed. Recent advancement of ontologies and semantic web, semantic specification language such as RDF and OWL is also highlighted (section 2.1)

Secondly, current state of the art of context awareness in manufacturing is reviewed with a focus on bridging the domain FMS with it. In this regard, nature of the FMS context model, principles of ontology-based FMS context modelling, needed functionalities for context management are discussed. Associated challenges are also reported (section 2.2).

Finally, this review presents the optimization requirements for modern FMS and the state of the art optimization techniques. The potentials of SOA-based control and the linking of emerging paradigms like context-sensitive computing with it, is also discussed (Section 2.3).

A conclusion of this review section is drawn in section 2.4. The organization of the review is depicted in figure 1.

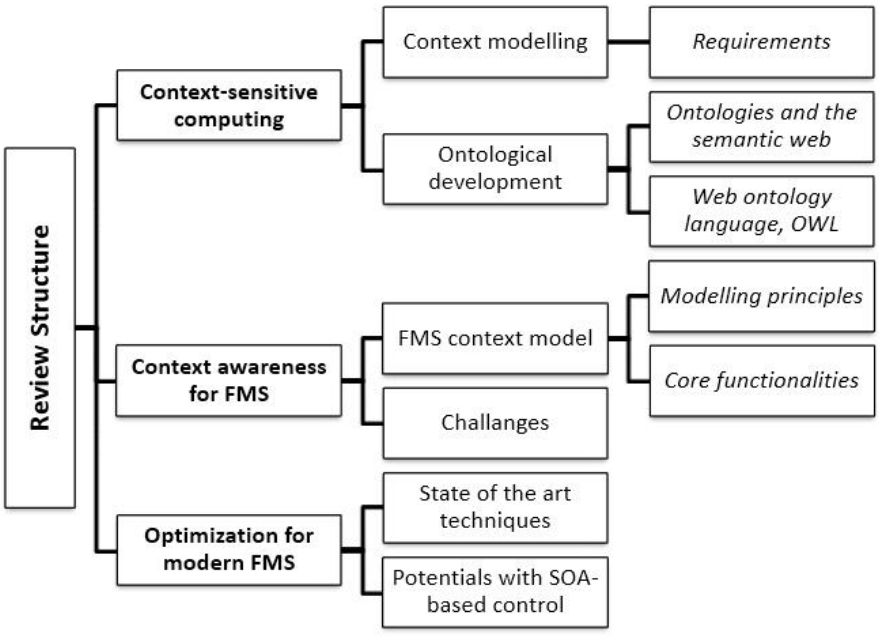


Figure 1. Organization of the literature and technology review.

2.1 Context-sensitive Computing

The term ‘context’ is defined as a mutual relationship between several conditions that exists in a given situation in which some actor exists or an event occurs [Schilit 1994]. The main constituents of a context are depicted in figure 2.

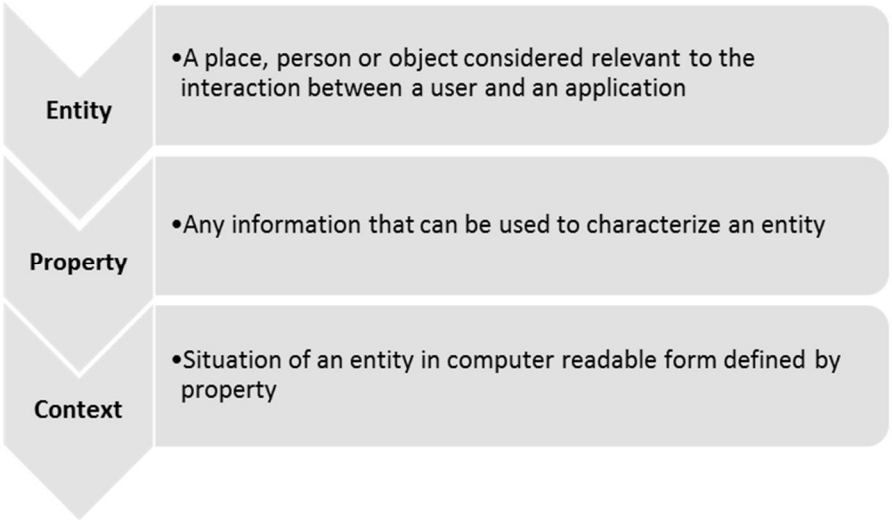


Figure 2. Constituents of a Context.

Contexts are primarily considered from two different views, user oriented and system oriented. In user orientation, context can be characterized through relationships that evolve around the user of a system which are of interest. At system orientation, context is any information that the system senses beyond any direct commands and which have an effect on the state of the system. Another group of researchers formulated the model of context as follows [Moore 2007]:

- A context describes a situation and the environment a device or a user is in.
- A context is defined by a unique name.
- For each context a set of features is relevant.
- For each relevant feature a range of values is determined (implicit or explicit) by the context.

The concept of context-sensitive computing is primarily propagated in the domain of ambient intelligence (AmI) and ubiquitous computing. The core concept defines the ability of computational entities to discover and to react to the environmental changes they are situated in. This is also understood as the capability of computational devices for identifying,

interpreting and responding to the environment from both user's and device's perspective. With the involvement of many researchers further, the definition has extended to several extents depending on the various application domains [Chen 2004].

Computational entities in context-depended applications can be both sensitive and reactive, depending on the environment. Context integrates various knowledge sources and binds knowledge to the user to ensure consistent understanding and this is the parallel reason for wide investigation of context awareness within knowledge management research. Such exemplary research on context sensitive computing can be classified primarily into two categories: context-based proactive delivery of knowledge and capture-utilization of contextual knowledge via support applications.

Context processing towards context-sensitive computing is typically categorized in three different ways [Gu 2005]. The first category deals with presentation of information and services to a user. The second category defines the automatic execution of a service in a more complex environment and the last one is tagging of context information for later retrieval (figure 3) [Moore 2007].

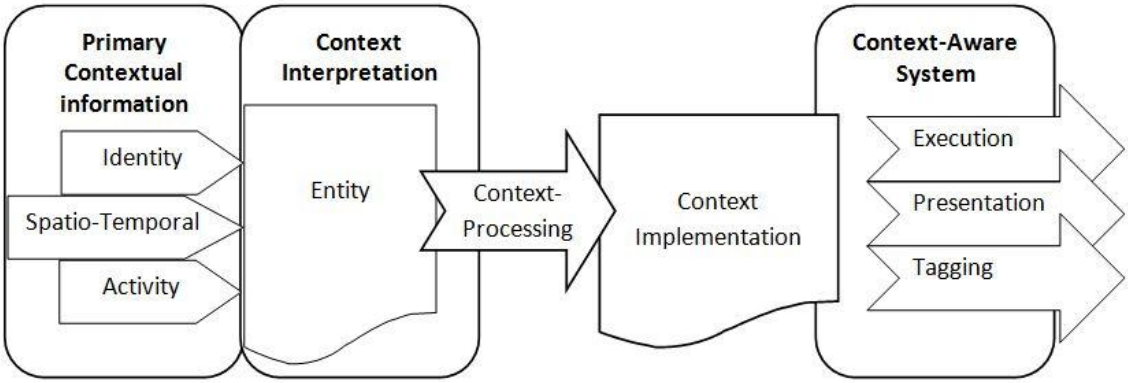


Figure 3. Context processing towards a context-aware system.

The notion of context sensitive computing, especially to achieve manufacturing process optimization, refers to process preferences of products and process skills of devices, the physical capabilities of equipment and environment conditions. However, similar computing for FMS to achieve the required level of optimization is challenging since an operating environment of modern FMS is highly dynamic and resides in distributed control.

2.1.1 Context Modelling

General methodology practiced in context-modelling are key-value models, Mark-up Scheme

Models, Graphical Models such as UML, Object oriented model (OOM), Logic-based Models and Ontology-based Models [Moore 2007]. Comparison of different context modelling techniques is reported by some researchers [Bettini 2010]. However, considering the level of formalism, distributed composition and applicability to existing environment and validation, the present research on context modelling is mostly focused on ontologies [Sattanathan 2006]. The recent advancement of ontologies in manufacturing offers the creation of a common language for sharing manufacturing knowledge among designers, design tools and software applications by providing a comprehensive semantic foundation of the facility [Uddin 2011] [Sandkuhl 2007].

2.1.1.1 Requirements

Some basic requirements [Bettini 2010] for context-sensitive systems are stated as follows.

- Heterogeneous data sources: The sources for contextual information, usually varies to a large extent depending on their extraction platform, update rate and their semantic level. Some sensor level data from the plant lower level often provide raw data (signals) that need to be further processed to utilize as a meaningful context [Zuraini 2010]. Most importantly, the update rate of such lower level devices is sometimes as fast as a fraction of seconds. On the other hand, contextual information from upper level, for instance the update rate from manufacturing execution system (MES) is relatively slow [Ge 2010]. Some contextual information might be more static like the resource information or planned production hours. This indicates that the sources of the contextual entities are heterogeneous; therefore a context model must be able to express those different types of contextual information per application's need.

- Dependency among contextual entities: A context model acts as the main data model for context-sensitive computing, which is utilized by various support applications. Therefore, the context model must extract the related entities to represent formally, ensuring an accurate behaviour of the domain. In doing so, the context model must also consider the dependent entities in a context model [Neovius 2006]. For example, if a particular instance of NC program changes in the context model, the dependent properties such as the NC program ticks also changes.

- Context historian: to utilize the context-sensitive computing for different application

usage, it is often needed to access the past states of different contextual entities. The past states are meant to be the reference context for mapping onto the currently extracted context. This is required to identify any changes of the required parameters based on the previous instances. Therefore, a context historian is required to be managed and updated with the reference context. In managing the context historian, the update rate of the reference context is often critical. It depends on the time required for context extraction, management and update on one hand, and the update rate required by the applications in real time on the other hand [Moltchanov 2009]. To maintain timeliness in context-sensitive computing, at least to answer the near real time application's need, the reference context should contain only the relevant entities in a refined context model stored in the historian.

- Efficient Context modelling and reasoning support: plant level data sources and data quality vary to a large extent depending on the devices and device level communication protocols. Often, the extracted contextual information might be incomplete. Therefore, an appropriate context modelling approach, suitable to the domain of interest is important [Verstichel 2008]. Contextual information may suffer from inconsistency and ambiguity as well. The reasoning support to the context model allows consistency checking and also provides ways to infer new explicit knowledge.

2.1.2 Ontological Development

Ontologies allow knowledge sharing, logic, inference and knowledge reuse and hence this is utilized for formal context representation and modelling across several domains. Ontology is "a formal explicit specification of a shared conceptualization" [Zuniga 2001]. Formal modelling through ontologies enables knowledge re-use and domain knowledge representation which are the basic needs for knowledge acquisition modules.

A shared context is referred to as ontology because the domain ontology offers a common understanding of the modelled concepts and of the explicit relations between them. Essentially, context ontology can be envisioned as close as of any other knowledge-representation systems. Each context contains a set of concepts that defines the basic terms which are then utilized to represent knowledge in the ontology. Furthermore, the constraints present in each context, controls the way how the instances of the concepts might be created and linked to other instances. In addition to these core functions, however, the role of context ontologies sets a number of further requirements on the representation language.

Several semantic specification languages such as RDF [Klyne 2004] and OWL [Schneider 2004] provide potential solutions for ontology-based context modelling (especially for the future pervasive computing environment where contextual information should be provided and consumed anywhere and anytime). RDF is a simple model supporting large-scale information management and processing, while considering different contexts from diverse sources. The assertions from sources can be united, providing additional information than they contain separately.

Significant research has been conducted to investigate the logical foundations of OWL and how this modelling language can be utilized to express a user's situation in various contexts [Luther 2005]. CONON [Wang 2004] is an OWL-based context ontology that allows logic-based reasoning in the modelled context. The RDF model for context reasoning in a pervasive computing environment, coupled with flexible context-based rules are presented in [Jari 2005] that recommends the available services with a priority order.

2.1.2.1 Ontologies and Semantic Web

The Semantic Web [Berners-Lee 2001] is characterized by an 'information web' which essentially differs in understanding in contrast to the current web. The main reason behind is the more usability of the semantic web by the machines than the current Web. Information on the Semantic Web remains in a structured form and defines an agreed-upon meaning. A similarity exists between a Semantic Web and a large online database in terms of containing structured information and most importantly providing an interface for queries. The information in a regular database in contrast, can be heterogeneous, which is not conforming to one single schema.

The primary standards within the semantic web are considered to be RDF (Resource Description Framework), SPARQL (SPARQL Protocol and RDF Query Language) and OWL (Web Ontology Language). RDF serves as the data modelling language, meaning the information in a semantic web is stored and represented as RDF. SPARQL provides the interface for various systems to query RDF data and OWL is the schema language [Klyne 2004].

Semantic web depends on ontologies for formal representation of the structured data, which remains at the core for machine understanding and associated communication [Brickley 2004]. Shareable domain ontologies enable both user and machine to communicate with each other to

support interchange of semantics. Therefore, development of ontologies, capturing domain specific concepts and linking of those is characterized as the core needs for semantic web [Hayes 2004] [Schneider 2004].

2.1.2.2 Web Ontology Language, OWL

In literature, Web Ontology Language (OWL) is defined as a language for knowledge representation for encoding ontologies in order to support the semantic web. OWL is a recommendation from W3C which has the compatibility with XML and with other W3C standards [W3C 2004]. OWL, which is an extension to RDF and RDF schema through additional vocabulary, allows formal representation of a particular domain. Formal representation is achieved by defining, for instance, the concepts or classes, their properties, relations between classes, cardinality, equity and enumerated classes within the domain ontology model [Deborah 2004]. OWL ontology is considered both as a valid RDF document and XML document syntactically. This allows OWL ontology processing via available XML and RDF-based tools.

At the implementation level, OWL has three sublanguages for defining the semantics, OWL-Lite, OWL-DL and OWL-Full. The former two semantics are built on Description Logics [Horrocks 2004]. Description logics have the expressiveness and meaningful computational properties, at the same time maintaining a computational completeness. OWL-Full utilizes a novel semantic model with an aim to provide RDF Schema compatibility. For a complete expressiveness, OWL-Full is adopted at user level, however, it has the associated computational complexity. Reasoning support for the full scope feature of OWL-Full is unlikely as expressed in [W3C 2004].

OWL-Lite is best suited for the users where the ontological usability requires hierarchical classification of the domain of interest and assigning simple constraints within the concepts. OWL-Lite is not adopted largely due to the limitation on expressiveness for complex constraints.

OWL-DL is intended for the maximum expressiveness for the ontology model and also ensures computational completeness. It provides the reasoning support for consistency checking utilizing the reasoning engines. Due to the correspondence of description logics, OWL-DL is named accordingly, which provides a formal OWL foundation.

Semantically, OWL-Full is different compare to OWL-Lite and OWL-DL. In OWL-Lite and OWL-DL, a resource cannot be defined as a class without formal description elsewhere in ontology document. However, the restriction is flexible in OWL-Full. Classes can be characterized as instances and unlike OWL-DL, it does not require to define explicitly the type of each resource and hence bringing extended expressiveness. However, most ontologies do not require this extensive expressiveness and hence OWL-DL is widely adopted [Heflin 2003].

The nature and the required outcome from the developed ontology generally indicate the sublanguage need for that particular model. The selection among OWL-Lite and OWL-DL varies to the extent of ontological expressiveness. The selection among OWL-DL and OWL-Full varies to the extent of meta-modelling and extended expressiveness requirements.

2.2 Context Awareness for FMS

Context-aware system and development of context sensitive support applications are relatively new in manufacturing. However, adoption of ontologies, as the core building block for context-sensitive computing reported in this work, is emerging in different areas of manufacturing and gaining a wide range of interest in recent years [Obitko 2008].

Manufacturing's Semantics Ontology, MASON [Lemaignan 2006] is a manufacturing ontology that describes a general purpose manufacturing semantics using OWL. It also highlights the usability of ontologies for formal representation and data sharing in manufacturing. An ontology addressing to mechatronic devices is developed by [Lopez 2006] that categorizes applicable hardware and software features in order to utilize the formalized knowledge in the automation domain [Vyatkin 2005].

Ontologies for logistic planning and ontologies addressed to the shop floor and reconfigurable assembly are examples within an agent-based manufacturing systems [Rzevski 2007]. Manufacturing ontologies offering shared manufacturing semantics enable the machines to communicate and bring transparency to complex devices, and hence contributing towards excellent manufacturing. Relevant research in this domain is mostly aiming for a seamless system integration to address the need for required interoperability between diverse systems [NIST 2010] [McLean 2005] [Zhou 2004].

In a distributed agent-based manufacturing environment, domain ontologies are utilized via knowledge sharing and re-use to gain process, product and system level information related to

status and control of the manufacturing process [Khedr 2004]. A formalized manufacturing model for FMS is reported by [Molina 1999], defining four level functionalities on the factory model, shop floor model, cell and station models. The developed model is aimed to provide a comprehensive semantics of the global manufacturing capability.

Ontology modelling applied OWL and OWL-S is reported by [Lin 2007] where an engineering, product development model enables inter-enterprise level communication and collaboration within different design teams. An ontology development approach using six steps is described by [Ahmed 2007], facilitating engineering design, together with research methods and assessment in each stage of the proposed approach. A formal representation of a product family using ontology in the semantic web paradigm is presented by [Nanda 2005]. The model allows hierarchical grouping of developed concepts for the relevant design objects, which eventually assists in product family design and reduces complexity, lead-time and development costs.

An ontology termed as DFM (design for manufacturing), is developed by [Chang 2010] to represent a manufacturing knowledge base for the facility. The aim is to share and re-use domain knowledge among the designers to assist in decision making for complex technical problems. DFM also supports in identifying data inconsistency and errors. A systematic ontological development is reported by [Lin 2011] addressed in a use case, electronic industry in order to provide support for engineering design [Uddin 2011].

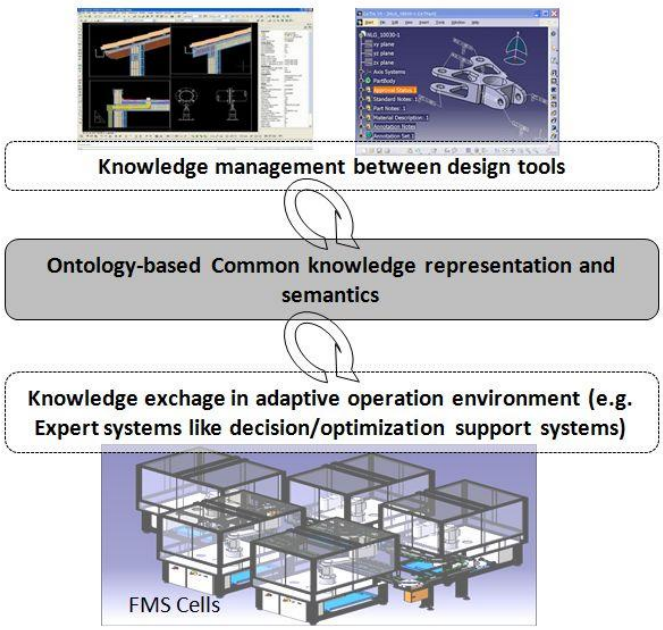


Figure 4. Utilization of ontology-based knowledge representation in manufacturing.

The main benefits of the recent progress of ontology research toward the implementation in manufacturing (figure 4), can be listed as follows:

- To create a common language to share knowledge about product, process and system among designers and support applications.
- To enable context-aware computing addressed in a complex, adaptive operation environment for decision support applications.
- To gain manufacturing knowledge to describe their structure and relations in a hierarchical manner.
- To share and to reuse manufacturing semantics and to infer new knowledge utilizing relations and axioms encoded in ontologies.
- To avoid extra overload of centralized software applications processing the raw data.

2.2.1 FMS Context Model

The purpose of the context model is to model the knowledge contexts relevant to product, process, device and resources in FMS. The aim is to further process KPI relevant entities from these extracted contexts. Typically the processed KPI relevant knowledge contexts are production orders, operation plans, device status and current job processing queue which has the influence on the global optimization. A typical production order taxonomy for FMS [Uddin 2012] is illustrated in figure 5.

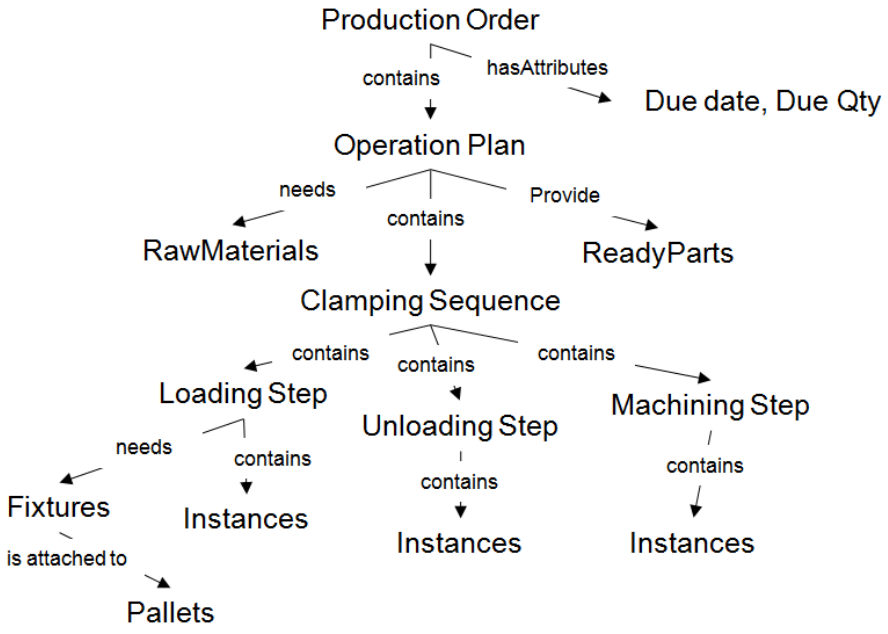


Figure 5. A typical FMS production order taxonomy.

2.2.2 Modelling Principles

Some basic principles of context modelling [Self-Learning 2012] in FMS are identified as follows:

- Support description of main context: In practice all context information is difficult to model and also not realistic. The context model, however, needs to consider the most relevant concepts and properties according to the requirement of support applications.

- Model the context that is easily acquirable: The concepts considered need to be identified clearly and integrated into the model effectively, whether fed automatically or by manual input explicitly [Baldauf 2007].

- Trade-off between the investment of context modelling, extracting and effects of context sensitive adoption: Generally, context modelling will be more accurate if a very detailed level capturing is done. However, the downside is that more time and effort is needed for detail level context capturing and processing, which has an impact on computational recourses in handling the detail level contexts. This has also the potential to bring deficiency to the run time optimization process.

2.2.3 Functionalities

The functionalities that are needed for context capturing followed by context management, are mainly relevant to the raw data monitoring from different plant level sources, extraction of contextual entities and identification of context sensitive information. Context management requires context reasoning to deduce knowledge context and context provisioning which provides the query interface for support applications. The functionalities mentioned are defined as follows [Self-Learning 2012] [Khedr 2004]:

2.2.3.1 Raw Data Monitoring and Context Extraction

To populate the context model with the raw data and to process it to infer high level contexts, relevant data is required to be monitored (e.g. from sensors, RFID, PLCs) and updated to the ontology model. The raw monitored data are then used for further processing and for context identification.

Contextual entities those are applicable to the support applications, need to be extracted during regular plant operation and to be further pushed for upper level processing. Figure 6 shows a

context extraction process at a conceptual level. The three main functionalities are context identification, context reasoning and context provisioning. From the raw monitoring data, context identification produces knowledge and relevant knowledge contexts. Context reasoning enables knowledge inference from low-level monitored context by means of reasoning engines [Chen 2008]. Reasoning on deduced context also provides consistency and reliability to the inferred knowledge contexts. Context provisioning allows contexts for optimization to realize intelligent and context-sensitive processing. The extracted context is to be stored in the context repository that serves the purpose of context historian.

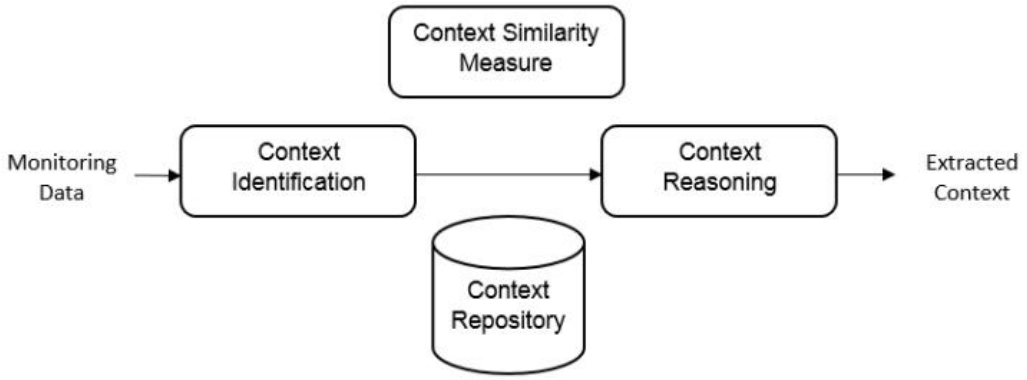


Figure 6. Conceptual context extraction process.

2.2.3.2 Context Identification

The conceptual context identification process is illustrated in figure 7. Context monitoring interfaces deliver as much context information as possible. Using the interfaces, a bunch of raw data can be extracted, for instance, from machines, from production orders for further context processing. The context identification process then maps the delivered data onto the ontology-based context model by means of an identified context.

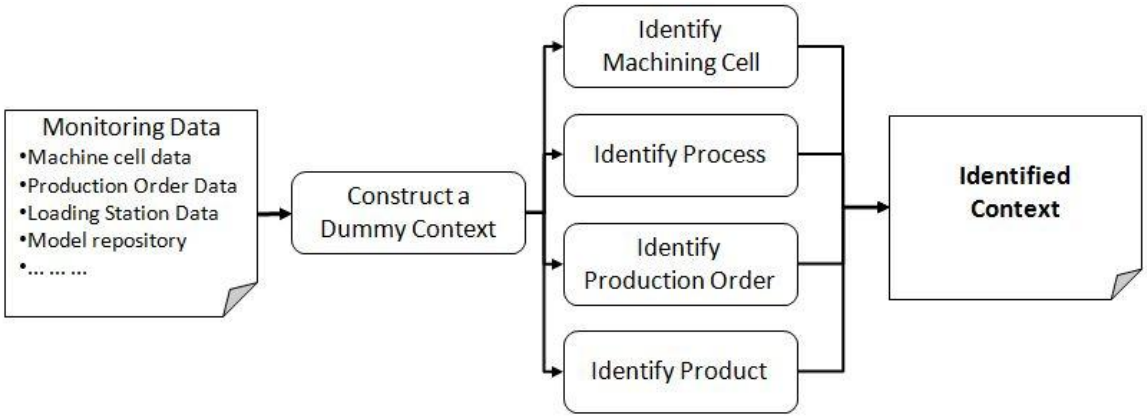


Figure 7. Conceptual context identification process

2.2.3.3 Context Reasoning

Identified context can be further processed using a contextual reasoning approach based on the formal description of contextual entities. Identified context can have some integral characteristics which may specify the contexts as incomplete, temporal and interconnected. Context reasoning can utilize the reasoning mechanisms to verify those characteristics, provide consistency and infer high level contexts from primarily extracted contextual information [Luther 2005]. Case studies related to ontological reasoning and the needs for solving such inconsistent context models through reasoning are reported by some researchers. The main focuses were proof checking, ontology validation and classification in the Protégé editor with RACER inference engine [Haarslev 2001].

A flexible approach for ontological reasoning can be achieved by encoding context-based rules or domain specific rules. Rule-based reasoning can be implemented for building prioritized inferred contextual knowledge and also for using of this knowledge accordingly per upper level application's need [Jari 2005].

Another method for context reasoning is to use deductive reasoning, which is a basic approach in logics. In this approach the knowledge inferences are implemented by using past known or identified facts. Deduction reasoning is a quite familiar methodology in general logics, and especially pertinent in logic programming. In addition, deductive reasoning allows consistency checking and improved reliability of the reasoned knowledge which could be fed in by incorrect monitoring. At ontology level, reasoning is possible depending on the semantics of the ontology language (e.g. OWL) and the definitions in the context ontology. Since, RDF and OWL are practiced to model context ontology, deductive reasoning is well supported in this regard [Qin 2007].

Application level reasoning is foreseen that uses the same deductive principles, but with application-specific rules (e.g. table 2). Conceptual reasoning with context sensitive information is illustrated in figure 8.

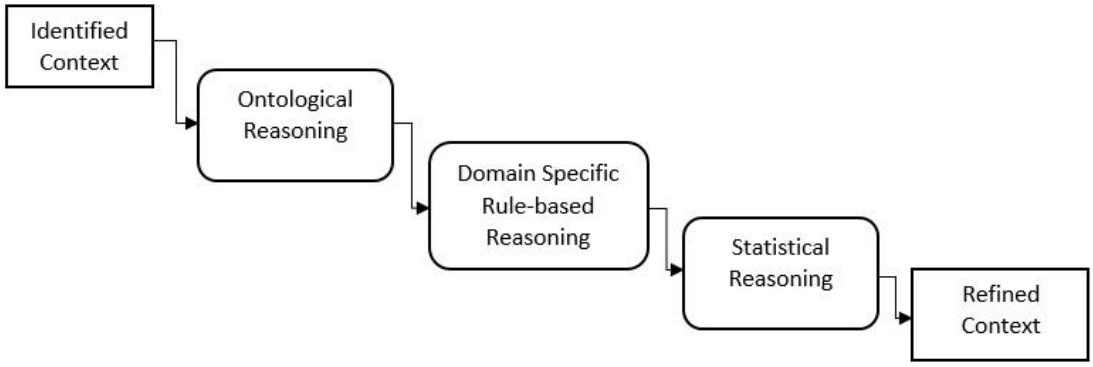


Figure 8. Context reasoning example

The approach for statistical reasoning does not rely on strict logical rules, instead it attempts to associate information into probable relations, as suggested by the empirical data. Statistical reasoning, however, allows the identification of the deduction rules, based on empirical context data. The techniques to perform such data-mining are relatively well-established [Srikant1997].

2.2.3.4 Context Provisioning

Context provisioning provides the domain specific reasoning and query interface for the support applications, e.g. OSS. Refined context to be provided proactively (proactive context provisioning) or depending on the request from support applications (on demand context provisioning). Since the semantic web ontology language OWL forms the foundation for context ontologies, SPARQL (an RDF query language and protocol) has potential to provide a good mechanism to support context provisioning [Hayes 2004].

SPARQL is similar to SQL language for querying RDF data. Structured SPARQL queries are formulated from various data sources to classify intended results. The queried data can be stored as RDF inherently or can be seen as RDF thru middleware. SPARQL has the capability to process queries per application's need and per optional graph patterns together with their conjunctions and disjunctions [Liu 2010]. Using the source RDF graph, SPARQL also allows extended value testing and putting constrained queries. The outcome of the SPARQL queries can be the result sets or RDF graphs.

2.2.4 Challenges

There are several associated challenges for dynamic context modelling, context capturing and context processing to fulfil the real time application's needs [Bettini 2010] [Zakwan 2010]. Plant floor sensor level data is variable and can vary frequently to a large extent as the quality

of the captured contexts can differ depending on the diverse range of sensor types. Therefore, context modelling approach must essentially support quality and the required richness level.

Contextual information may suffer from incompleteness and ambiguity as well. Context modelling and processing must incorporate the capability to handle these issues by interpolation of incomplete data on an instance level [Huang 2004]. However, the description of contextual facts and interrelationships in a precise and traceable manner represents a significant challenge. It is important that the context model can be adapted to enable the use of the model in existing domains, systems and infrastructures. The context model should also be re-usable so that it can be utilized across other similar domains.

2.3 Optimization for Modern FMS

Optimization research has been a wider research topic across diverse manufacturing domains for many years. In FMS, optimization in manufacturing systems, operations, costs, scheduling are some of the active research fields. Research on FMS optimization, especially in addressing the run time optimization need, however, is comparatively new and gaining interest in recent times [Cao 2008].

An FMS is generally known to be as an integrated and computer controlled system associated with automatic material handling stations and processing stations like machine tools and devices. The control system and the different stations are typically synchronized in a complex way to respond the simultaneous processing need of different volumes and product ranges [Stecke 1983]. The FMS brings the flexibility to integrate production line efficiency to the facility of a job shop in order to accommodate batch production aiming moderate product volume and variety. However, there are associated costs for gaining flexibility and the required capital investment is typically very high as well. Therefore, careful attention is needed for the proper planning of an FMS during the design and development stage. Before production, a through operational planning is important in order to identify the system's efficiency over time. Hence, the detail planning compared to other conventional production paradigms is the key for a successful operational FMS.

The job processing relevant decisions within FMS operations falls mainly in pre-release and post-release phases. Operational planning related to pre-arrangement of parts, jobs and tools, expected schedule, machine utilization, downtime, for instance, is associated with pre-release phase. Post-release phase mainly indicate the addressing of dynamic scheduling problems due

to run time change of priorities, sudden machine breakdown and relevant facts. Pre-release phase and required decisions in this phase, for instance, machine grouping, job selection, production throughput, resource distribution and loading problems are relevant to the setting up an FMS as well [Kim 1998]. Machine loading, among other factors, is one of the most critical production planning problems due to its direct impact on the performance of FMS. Specifically, loading problems within FMS refers to job allocation to different work stations considering different constraints, with an aim of fulfilling various performance objectives.

Significant research has been conducted by researchers to obtain effective solutions to loading problems and also minimize the computation burden at the same time. Different mathematical models, heuristics and meta-heuristics-based approaches, using simulation models are some of the widely adopted methodologies in this regard [Stecke 1983].

Post release decisions are critical in modern FMS plants. FMS plants need to deal with numerous challenges like parallel jobs processing, buffer allocation, highest machine utilization, minimizing production lead times, maintaining due delivery date, responding to unexpected events and minimizing tool flow. These factors influence the overall factory throughput. Job scheduling problems are of main interest as the production orders and job processing priority change dynamically to meet the production order lead time or to maintain the highest machine utilization rates. Several priority rules used in manufacturing plants are listed in table 2.

Table 2. Typical priority production rules in FMS.

Abbreviation	**Priority Rule**
FIFO	First in First Out
FOFO	First Off First On
SPT	Shortest Processing Time
LPT	Longest Processing Time
SRPT	Shortest Remaining Processing Time
LRPT	Largest Remaining Processing Times
EDD	Earliest Due Date

In a planned or pre-released scheduling, an optimal job dispatching queue is generated with the available jobs and usually simulation methods are utilized. A planned model provides a good basis for resource planning and can be used to predict the optimum. But, run time changes and unexpected events make the planned schedule ineffective. Therefore, optimal FMS operation considering reactive scheduling (post release decision) is of the main interests of the

manufacturers. Hence, the modern FMS utilizes complex and adaptive control systems, which promotes integration of various decision support applications.

Context-sensitive decision support systems or optimization support systems deals with the context model of process, product and system to identify contextual changes. It also requires contextual mapping with formally represented manufacturing knowledge to enable knowledge inference by the support applications. One of the main research motivations to integrate context sensitive client applications to the FMS controls is to deal with the post release decision support in an adaptive operational environment to ensure optimality.

2.3.1 State of the Art Techniques

A wide range of approaches are adopted in industrial and research level to address optimization in manufacturing. Optimum seeking algorithms (e.g. Branch and Bound search, dynamic programming), heuristics/meta heuristic algorithms (genetic algorithm, simulated annealing), simulation models and are mostly used [Kumar 2006] [Uddin 2010]. Different optimization techniques in manufacturing utilized at research and application level are depicted in figure 9.

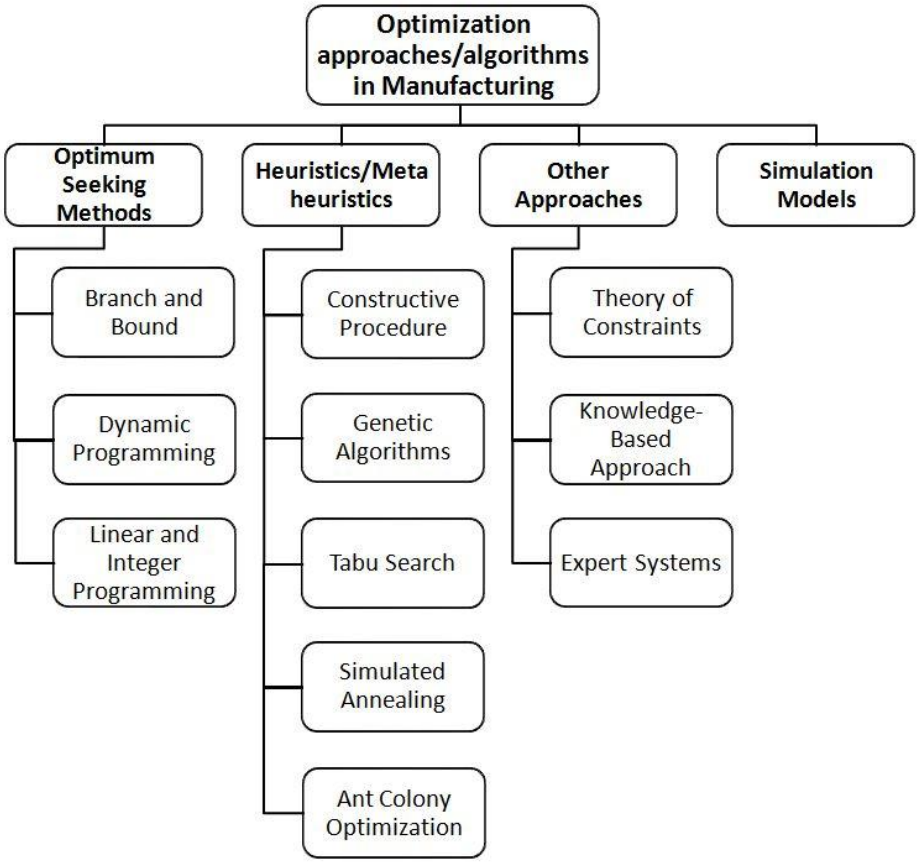


Figure 9. Different optimization techniques in manufacturing.

The theory of constraints, knowledge-based approaches and expert systems (e.g. agent-based) are emerging and have gained a lot of interests, especially in the development of high performance microprocessors utilized as intelligent entities embedded in the plant floor infrastructure.

2.3.2 Potentials with SOA-based Control

The possibility to incorporate intelligence, even in the smaller devices via high performance microprocessors has made possible knowledge-based, semantic web service- enabled manufacturing [Brenan 2004]. SOA deployed by Web Services (WS) has been recognized an answering the needs of a highly reconfigurable system: loose-coupling and dynamic discovery of new processes [Lastra 2006], fulfilling the need of dynamic optimal decision making.

Traditional enterprise application technologies as Distributed Computing Environment (DCE), Common Object Request Broker Architecture (CORBA), Microsoft's Distributed Component Object Model (DCOM), Java 2 Enterprise Edition (J2EE) are lacking explicit platform-independence due to their use of specific sets of communication standards and protocols. The integration of critical applications is within reach due to the adoption of WS and SOA. WS are currently supported by all major independent software vendors, including platform vendors such as IBM, Microsoft, SAP, PeopleSoft, Oracle, Sun, and BEA. Tool support for WS and related technologies is growing [Shirley 1992] [OMG 1996] [DCOM] [Grosso 2001].

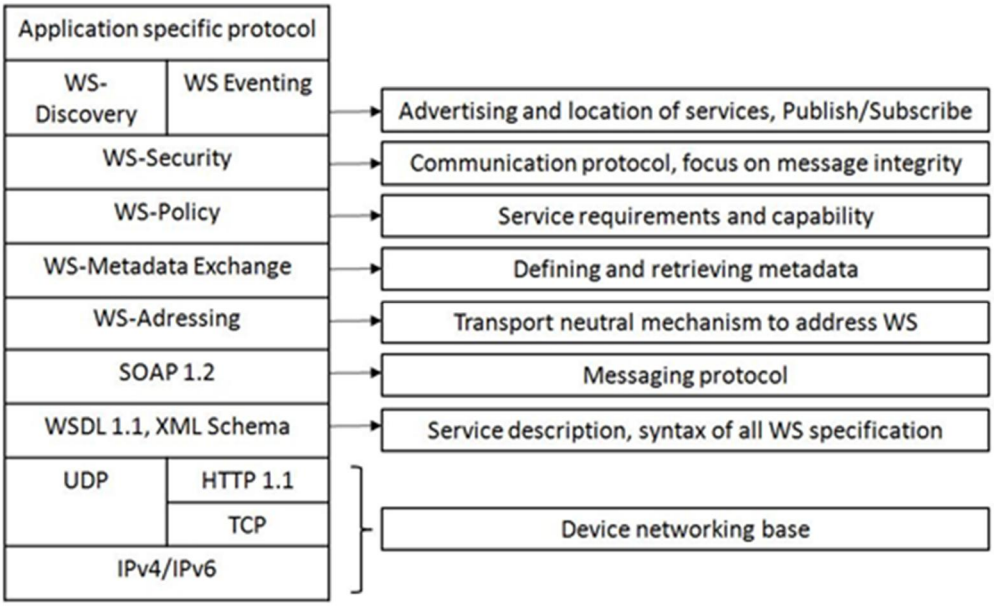


Figure 10. DPWS protocol stack.

Although in the software world SOA and WS are already widely adopted, SOA-compliant manufacturing is still an emerging paradigm. The drawbacks for several device-level SOA integration technologies such as [Jini] and [UPnP] are: lack of platform neutrality, lack of adaptation to resource restricted devices and specific protocols for device discovery/eventing.

Device Profile for Web Services (DPWS) is an extension of the Web Services protocol suite that defines the minimal set of implementation constraints to enable secure WS description, messaging, and dynamic discovery, publish/subscribe eventing at device level (figure 10).

DPWS is equipped with WS standards (e.g. WSDL, XML schema, SOAP, WS-Addressing, WS-Metadata Exchange, WS-Transfer, WS-Policy, WS-Security, WS-Discovery and WS-Eventing) that facilitate ideal integration at application, process and device level, application interoperability and re-use of IT assets.

At present, DPWS is considered to be the most widely practiced technology for implementation of SOA-compliant production systems. Pilots of DPWS-enabled devices in the industrial domain [SOCRADES 2009], [SODA 2008], [SIRENA 2005] are considered to be the first step towards achieving both horizontal collaboration and vertical integration.

A scalable SOA deployed by WS has potential to achieve flawless integration, interoperation and required flexibility. This allows detection and interpretation of data from existing database systems, device level, data servers, and file systems, which include plant specific process, equipment, enterprise information mostly XML files, NC programs (text files) and digital/analogue signals from sensors.

2.4 Conclusions

Existing state of the art technologies relevant to context modelling and further context processing in manufacturing domains differ in the expressive power of the context information models, the support they can provide for reasoning, computational performance of reasoning and the nature of the application domain.

As a recent trend in modern manufacturing, industries tend to seek continually for higher productivity at an optimum efficiency and cost reductions by using real time data and integration of internet of things (IoT) to the industrial value chain. This trend also serves as the foundation for the next generation industries, i.e. Industry 4.0 [Kagermann 2013]. Among other design principles of Industry 4.0 [Herman 2016], information transparency and interoperability

remains at the core for the smart factories where the aim is to aggregate raw monitor data to higher level context information and to process contextual entities for the machines, devices and people to communicate with each other.

In this literature review, current state of the art techniques are presented as the enabler of context sensitive computing in a dynamic environment and capitalization of ICT for optimizing the future FMS.

3 Context-sensitive Optimization for FMS

This chapter presents five peer-reviewed publications related to this thesis from different perspectives. Publication I contributes to the optimization technique by focusing optimal line balancing and sequencing for a mixed-model assembly line (AL). Production execution in an AL requires many important factors to be considered for optimization. Different line orientations, production approaches, line characteristics, performance and workstation indexes define objective functions for optimal line balancing and product sequencing. This paper analyses important AL design characteristics and also provides an integrated approach for balancing of mixed-model assembly lines (MMALs) combined with optimal product sequencing.

Publication II presents an ontology-based knowledge representation for FMS, providing a comprehensive semantic foundation of the facility. The domain ontology model that is addressed captures and formally represents the manufacturing semantics from heterogeneous data sources, allowing knowledge sharing, re-use and update.

Publication III presents how service oriented architecture (SOA) and supporting technologies can be bridged together with the emerging production paradigms to meet the required level of flexibility, interoperability and communications.

Publication IV presents a context-sensitive computing approach, integrated with an SOA-based FMS control platform. This approach addresses how to extract manufacturing contexts at source, how to process contextual entities by developing an ontology-based context model and how to utilize this approach for real time decision making to optimize the key performance indicators (KPIs).

Finally, publication V presents an application of context-sensitive optimization for FMS, considering the dynamic machine utilization rate and overall equipment effectiveness (OEE) as the key performance indicators (KPIs). Runtime contextual entities are used to monitor KPIs continuously to update an ontology-based context model and subsequently convert it into business-relevant information via context management. The delivered high-level knowledge is further utilized by an optimization support system (OSS) to infer optimal job (re) scheduling and dispatching, resulting in a higher machine utilization rate at run-time.

3.1 Optimization for Assembly Line-based Manufacturing (Publication I)

The present global market environment is competitive and rapidly growing. Major manufacturers are trying to cope up with this changing scenario by optimizing their manufacturing design process. Modern discrete assembly-based product industries are associated with assembly lines (ALs) for greater efficiency and flexibility. Application of ALs was adapted for high volume, low-variation mass production in its initial phase. However the changing business world where the demand is mostly customer driven, has motivated the manufacturers to implement assembly-based manufacturing for job shop and batch production to create greater product variability. Mixed-model assembly lines (MMALs) facilitates product variations and diversities on the same line in an intermixed scenario. Hence, optimal AL design, balancing and product sequencing of mixed-model assembly are the major challenges for manufacturers for creating high-variety and low-volume product within the layout process. Different demand scenario, performance objectives, product mix, AL orientations, manual/robotic or hybrid workstation indexes, design constraints and performance indexes all play a substantial role in AL-based industries.

This paper identifies the most important AL design characteristics in its initial phase. Proper acknowledgement and association of these parameters with AL design, balancing, and mixed-model scheduling facilitates optimal solutions for improving overall line efficiency. In later phase of the paper, an integrated approach for balancing and sequencing of MMALs of problem type 2 (Boysen et al., 2007) is developed to optimize shift time for mixed-models with a predefined number of workstations considering smoothed station assignment load (SSAL) for job shop production. The approach presented in this paper also determines a smooth production schedule through optimal product sequencing.

3.2 How to Utilize Knowledge for FMS (Publication II)

In FMS, the plant operations consider several objectives like keeping the due delivery date of production orders, minimizing the production order lead time and maximizing the machine utilization rates. These objectives are usually adjusted dynamically depending on the production profile, process state and shift models. At the same time, the need for parallel, distributed jobs processing in optimal condition poses significant challenges to modern FMS plant operations. In response, complex, decentralized and adaptive control architectures are

utilized to address the run-time events occurring at the lower factory level, which promote integration of various decision support applications. Knowledge-based optimization support systems for run-time critical decision-making are the recent derivative terms, which need to deal with machine readable manufacturing semantics to apply knowledge inference. Semantic technologies and knowledge-intensive manufacturing is a growing research areas and Knowledge Base (KB) and inference engines remain at the core of such research. At the same time ontologies are considered as the catalyst for formal Knowledge Representation (KR), sharing and mediation in distributed environments.

The work addressed in this manuscript provides a comprehensive semantic foundation within an FMS domain, enabling knowledge representation and knowledge exchange through support applications. An ontology-based KR is addressed, providing an interpretation of the modelling elements in web ontology language (OWL). The aim is to enable precise real-world semantics of the FMS facility (production orders, products and resources) allowing knowledge sharing and re-use by disparate client applications. The semantic foundation also addresses the integration and the update of run-time process information to the OWL ontology model to support adaptive client applications.

3.3 How to Encapsulate and Re-use Production Knowledge via SOA (Publication III)

Technologies, leveraging artificial intelligence at the factory floor, knowledge-based system development and machine learning are being studied to make capabilities of self-X properties like self-adaptation, self-optimization and self-maintenance available to production systems.

This manuscript addresses a Self-Learning production system, which is a new concept to apply cybernetic principles to derive intelligent production systems. The system self-adapts and learns in response to the dynamic updates in contextual entities extracted from all factory levels. The context awareness approach addresses the integration of control and maintenance processes for necessary adaptation, which improves the transparency of complex processes and the overall equipment effectiveness especially regarding system availability and productivity. A reliable and secure software service-based integration infrastructure using distributed networked embedded services in the device space is the key to achieve such system.

Presenting the architecture of Self-Learning production system, this paper mainly analyses

how SOA as an architectural paradigm and supporting technologies can be bridged together to achieve a seamless enterprise wide connectivity using flexible, loosely coupled and reusable services.

3.4 Ontology-based Context-sensitive Computing for FMS (Publication IV)

The work carried out in this publication addresses an ontology-based approach to collect context-sensitive information from heterogeneous data sources to provide support for optimization in FMS. An ontology-based context model is developed to collectively generate and manage manufacturing knowledge and utilize it for real-time optimal decision making. Firstly, this paper presents a brief review of the relevant state-of-the-art technologies. Context modelling using ontologies, bridging of the ontologies to the semantic web paradigm, ontology modelling language and applications of ontology in manufacturing are addressed. Secondly, this work presents a context-sensitive computing approach to FMS addressing the needs for run-time optimization. Ontology-based context modelling and context processing through context identification, context interpretation, reasoning and context provisioning are discussed as the fundamental requirements of the presented approach.

A framework for a context-sensitive optimization support system is proposed, integrated on top of the generic FMS control platform with an aim of context-based manufacturing knowledge delivered to the support applications.

Finally, this paper considers a practical FMS use case, utilizing a service-oriented architecture (SOA)-based control paradigm, as an implementation platform. Defining the system architecture in brief, the implementations of the lower level functionalities of ontology-based context modelling, interfacing to SOA platform and context extraction from run-time raw monitored data are reported.

3.5 Context-sensitive Optimization of the KPIs for FMS (Publication V)

Dynamic job (re) scheduling and dispatching are becoming an essential part of modern FMS plant controls in addressing the chaotic nature of the production environment. Process plans in FMS need to consider several factors at run-time like demand fluctuations, extreme product customisations and run-time priority changes. To meet this plant-level dynamism, complex control architectures are utilized to provide an automatic response to such unexpected events. These run-time responses from the control systems deal with final moment change of the control parameters, which eventually influences the key performance indicators (KPIs) such as machine utilization rate and overall equipment effectiveness (OEE).

Execution of the plant level objectives (e.g. Keeping a higher machine utilization rate at run-time) in a chaotic job processing order is hence becoming crucial.
As mentioned in section 3.4, Publication IV reports the lower level implementation to the SOA-based FMS use case to provide the functional requirements of extracting standardized monitoring data from use-case web services (WS). The focus is to convert raw monitoring data to web ontology language (OWL) instances, populating and updating of dynamic entities to an ontology-based context model. In publication V, the work reports detail features and functionalities of higher level components in achieving the overall context-sensitive optimization for FMS, focusing on the run-time device-level KPI optimization. The reference architecture is elaborated by introducing higher level components for context management and their functionalities. An optimization support system (OSS) is addressed, where an optimization algorithm consumes contextual changes specified to the KPIs and suggests optimal job (re) scheduling and dispatching, maintaining a higher machine utilization rate at run-time.

3.6 Summary

The main contribution presented in this thesis is illustrated in figure 11, resulting in a new approach for runtime optimization methodology at the KPI level in a dynamic operating environment of FMS based on context-sensitive computing.

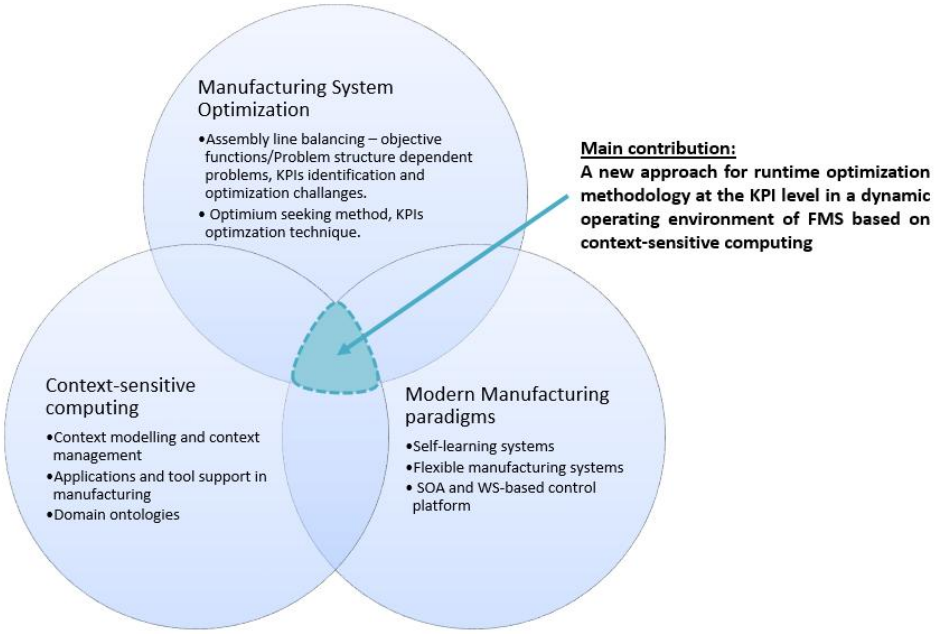


Figure 11. Main contribution area of the thesis.

Below table 3 summarizes the main results from the listed five publications in this thesis that contribute to this novel application of context-sensitive computing to FMS optimization.

Table 3. Main results as an outcome of this thesis.

Publication	Research method	Results
Publication I Optimization for AL-based manufacturing	Methodology for balancing mixed-model assembly lines (MMALs) combined with sequencing heuristic for optimal line balancing and product sequencing.	Optimization in the manufacturing process, identification of KPIs that directly influences the plant operations.
Publication II How to utilize knowledge for FMS	Ontology-based knowledge representation: domain ontology model development for FMS that captures and formally represents the manufacturing semantics from heterogeneous data sources allowing knowledge sharing, re-use and update.	A comprehensive semantic foundation enabling the plant operations to become knowledge intensive, distributed and collaborative in nature. This also facilitates knowledge management among various design tools and knowledge exchange in an adaptive operation environment through decision support applications.

Publication III How to encapsulate and re-use production knowledge via SOA	Bridging of adaptive production techniques and SOA as an architectural paradigm with an aim to achieve a seamless enterprise wide connectivity using flexible, loosely coupled and reusable services.	Application of SOA-based technologies to the lower factory level in achieving knowledge intensive and adaptive production system.
Publication IV Ontology-based context-sensitive computing for FMS	Integration of context-sensitive computing, on top of an existing SOA-based FMS control platform.	An approach addressing how to extract manufacturing contexts at source, how to process contextual entities by utilizing an ontology-based context model and how to implement this approach for real time decision making
Publication V Context-sensitive optimization of the KPIs for FMS	Context-sensitive optimization for SOA-based FMS, considering the dynamic machine utilization rate and overall equipment effectiveness (OEE) as the key performance indicators (KPIs).	A common interface for context acquisition and context management to deduce high level knowledge from raw monitored data. An optimization support system (OSS), where an optimization algorithm consumes contextual changes specified to the KPIs and suggests optimal job (re)scheduling and dispatching, maintaining a higher machine utilization rate at run-time.

4 Conclusions and Recommendation for Future Works

The operating environment of FMS is highly domain specific which in turn puts significant challenges to set unified optimal conditions. End users of FMS utilize different run time KPIs such as higher machine utilization rate, optimal production order lead time, alignment of due delivery dates, minimize the tool flow or a combination of them.

Moreover, these optimization objectives (KPIs) are dynamic. Their changes depends not only on the production profile, but also on the process state and shift model. For example, when most of the production load is addressed for direct customer orders it is necessary to keep the due dates as a priority. When most of the production is for Kanban manufacturing (stock batches) then the highest priority is machine utilization. This is the parallel rationale that the KPIs cannot be compared to any specified value, since those are mostly context dependent.

In this thesis, the proposed context-sensitive optimization demonstrates its effective integration within an existing FMS control platform, aiding the shop floor managers/operators in their dynamic decision-making process.

The result of this work shows, how interoperable contextual knowledge of the presented architecture can be used to address optimization of KPIs like machine utilization rate and OEE, while considering the chaotic job processing nature of FMS plants.

Exploitation of the addressed context-sensitive optimization enables the users to understand that the system supports to decide the optimal production rule and subsequent job scheduling which ensures maximum machine utilization at that particular contextual situation.

The reference architecture and associated modular development for the use-case implementation shows that formally modelled knowledge context and subsequent processing of those can be productively used in run time decision making through support applications.

4.1 Concluding Remarks

The objectives of this work have been achieved by enabling the solution of context-sensitive computing to the dynamic environment of FMS, in optimizing runtime KPIs for global factory optimization. This brings the bridging of two emerging domains of applied research which are context awareness and optimization research on FMS. The results obtained from practical SOA-based FMS use case validation also provides the targeted outcome from this research.

Therefore, this thesis contributes to the advance of the state of the art in the field of context aware manufacturing plant within the defined scope.

4.2 Potential Enhancements and New Research Directions

The functional implementation of the overall approach to the practical FMS use-case also provides a motivation for the further extension of this research. In the future, this research can be extended to incorporate a learning behaviour to the OSS, based on the operator's actions to the suggested optimization proposals. Such learning behaviour requires the operator to give feedback to each suggested optimal condition. Otherwise the system does not get right data for learning and learning-based proactive decision making. Data mining tools can be integrated to identify the contextual patterns of the operators' decisions, which can eventually be the feed for further KPIs identification and research. These patterns can be used to provide prior knowledge to the OSS.

References

[Ahmed 2007] Ahmed, S., Kim, S. and Wallace, K. M., "A Methodology for Creating Ontologies for Engineering Design", Journal of. Computer Information Science, vol. 7, no. 2, pp. 132-141, 2007.

[Baldauf 2007] Baldauf, M., Dustdar, S. and Rosenberg, F., "A survey on context-aware systems", International Journal of Ad Hoc and Ubiquitous Computing, vol. 2, no. 4, pp. 263-277, 2007.

[Berners-Lee 2001] Berners-Lee, T., Hendler, J. and Lassila, O., "The semantic web", Scientific American, vol. 284, no. 5, pp. 34-43, 2001.

[Bettini 2010] Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A. and Riboni, D., "A survey of context modelling and reasoning techniques, Pervasive and Mobile Computing", Science Direct, Vol. 6, No. 2, 2010, Pages 161-180, ISSN 1574-1192.

[Brenan 2004] Brennan, R. W., Christensen, J. H., Martinez Lastra, J. L., Martell, A. and Vyatkin, V., "OOONEIDA Community - a Prototype of an Open Object Oriented Knowledge Economy for Intelligent Industrial Automation", International IMS Forum, Villa Erba, Cernobbio, IT, 2004.

[Brickley 2004] Brickley, D. and Guha, R. V., "RDF Vocabulary Description Language 1.0: RDF Schema", W3C Recommendation: http://www.w3.org/TR/rdf-schema/, 2004.

[Cao 2008] Cao, Y., Xu, X. and Sun, F., "Data mining for the optimization of production scheduling in flexible Manufacturing System", 3rd International Conference on Intelligent System and Knowledge Engineering, pp. 252-255, ISKE 2008.

[Chang 2010] Chang, X. , Rai, R. and Terpenny, J., "Development and Utilization of Ontologies in Design for Manufacturing", Journal of. Mechanical. Design, vol. 132, no. 2, 2010.

[Chen 2004] Chen, H., Perich, F., Finin, T. and Joshi, A., "SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications", In Int. Conf. on Mobile and Ubiquitous Systems: Networking and Services, 2004.

[Chen 2008] Chen. Z., Liu, S. and Wang, X., "Application of context-aware computing in Manufacturing Execution System", IEEE International Conference on Automation and Logistics, pp. 1969-1973, Qingdao, 2008.

[DCOM] DCOM (Microsoft's Distributed Component Object Model) available online at, http://research.microsoft.com/en-us/um/people/ymwang/papers/html/dcomncorba/s.html.

[Deborah 2004] Deborah L. M., and Harmelen, F. V. (Editors), “OWL Web Ontology Language Overview , W3C Recommendation”, 10 February 2004, http://www.w3.org/TR/2004/REC-owl-features-20040210/.

[Dickinson 2009] Jena Ontology API, Available online at https://jena.apache.org/documentation/ontology/.

[Ge 2010] Ge, Z., Liao, Q. and Li, T., “A context-aware access control model for manufacturing grid”, International Conference on Mechanic Automation and Control Engineering (MACE), pp. 543-546, Wuhan, 2010.

[Grosso 2001] Grosso, W., “Java RMI”, O'Reilly, ISBN: 1-565-92452-5, 2001.

[Gu 2005] Gu, T., Pung, H. K. and Zhang, D. Q., “A service-oriented middleware for building context-aware services”, Journal of Network and Computer Applications, vol.28, no.1, pp.1-18, 2005.

[Haarslev 2001] Haarslev, V. and R. Moller. RACER system description. Siena, Italy: Springer-Verlag, 2001.

[Hayes 2004] Hayes, P., “RDF Semantics”, W3C Recommendation: http://www.w3.org/TR/rdf-mt/, 2004.

[Heflin 2003] Heflin, J., “OWL Web Ontology Language Use Cases and Requirements”, W3C Proposed Recommendation 15 December 2003. Available online at: http://liris.cnrs.fr/amille/enseignements/Ecole_Centrale/projets_2004/owl.pdf.

[Herman 2016] Hermann, M., Pentek, T. and Otto, B., "Design Principles for Industrie 4.0 Scenarios," 2016 49th Hawaii International Conference on System Sciences (HICSS), Koloa, HI, 2016, pp. 3928-3937.

[Horrocks 2004] Horrocks, I. and Schneider, P. H., “Reducing OWL entailment to description logic satisfiability”, Sanibel Island, FL, United States: Elsevier, Amsterdam, 1000 AE, Netherlands, 2004.

[Huang 2004] Huang, W. and Webster, D., “Enabling Context-Aware Agents to Understand Semantic Resources on The WWW and The Semantic Web”, Proceedings. IEEE/WIC/ACM International Conference on Web Intelligence, pp. 138- 144, 2004.

[IIOP] IIOP (Internet Inter-ORB Protocol) available online at http://www.omg.org/library/iiop4.html.

[Jari 2005] Jari, F., Ora, L. and Tapio, S., "RDF-Based Model for Context-Aware Reasoning in Rich Service Environment", in Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops, 2005.

[Jini] The Community Resource for Jini technology, available online at http://www.jini.org.

[Kagermann 2013] Kagermann, H., Wahlster, W. and Hekbik, J., "Recommendations for implementing the strategic initiative INDUSTRIE 4.0", Avaialable online at: http://www.acatech.de/

[Khedr 2004] Khedr, M. and Karmouch, A., "ACAI: agent-based context-aware infrastructure for spontaneous applications", Journal of Network and Computer Applications, vol. 28, no. 1, pp. 19-44, 2004.

[Kim 1998] Kim, Y-D. and Jeong, K., "A Real-Time Scheduling Mechanism for a Flexible Manufacturing System: Using Simulation and Dispatching Rules", International Journal of Production Research, vol. 36, no. 9, pp. 2609 – 2626, September 1998.

[Klyne 2004] Klyne, G. and Carroll, J. J., "Resource Description Framework (RDF): Concepts and Abstract Syntax 2004", W3C Recommendation: http://www.w3.org/TR/rdf-concepts/, 2004.

[Kumar 2006] Kumar, A., Prakash, M.K. Tiwari, R. S. and Baveja, A., "Solving machine-loading problem of a flexible manufacturing system with constraint-based genetic algorithm", European Journal of Operational Research, vol. 175, no. 2, pp. 1043-1069, ISSN 0377-2217, December 2006.

[Lastra 2006] Martizez Lastra, J. L. and Delamer, I., "Semantic Web Services in Factory Automation: Fundamental Insights and Roadmap", IEEE Transaction on Industrial Informatics, vol. 2, no. 1, pp. 1-11, 2006.

[Lemaignan 2006] Lemaignan, S., Siadat, A., Dantan, J. Y. and Semenenko, A., "MASON: A proposal for an ontology of manufacturing domain", In: IEEE Workshop on Distributed Intelligent Systems, pp. 195–200, 2006.

[Lin 2007] Lin, K. and Harding, J. A., "A manufacturing system engineering ontology model on the semantic web for inter-enterprise collaboration", Computers in Industry, vol 58, no. 5, pp. 428-437, ISSN 0166-3615, 2007.

[Lin 2011] Lin, L.F., Zhang, W. Y., Lou, Y. C., Chu, C. Y. and Cai, M., "Developing manufacturing ontologies for knowledge reuse in distributed manufacturing environment", International Journal of Production Research, vol 49, no. 2, pp. 343 – 359, 2011.

[Liu 2010] Liu, C., Wang, H., Yu, Y. and Xu, L., "Towards efficient SPARQL query processing on RDF data", in *Tsinghua Science and Technology*, vol. 15, no. 6, pp. 613-622, Dec. 2010.

[Lopez 2006] Lopez, O. and Martinez Lastra, J. L., "Using Semantic Web Technologies to Describe Automation Objects", International Journal of Manufacturing Research, vol. 1, no. 4, pp 482–503, 2006.

[Luther 2005] Luther, M., Mrohs, B., Wagner, M., Steglich, S., Kellerer, W., "Situational reasoning - a practical OWL use case", Autonomous Decentralized Systems, 2005. ISADS 2005. Proceedings, vol., no., pp.461, 468, 4-8, 2005.

[Luther2005] Luther, M., "Situational reasoning - a practical OWL use case", Chengdu, Jiuzhaigou, China: IEEE, 2005.

[McLean 2005] McLean, C., Lee, Y., Shao, G. and.Riddick, F., "Shop Data Model and Interface Specification", NISTIR 7198, National Institute of Standards and Technology, 2005.

[Molina 1999] Molina, A. and Bell, R., "A manufacturing model representation of a flexible manufacturing facility", Proceedings of the Institution of Mechanical Engineers, Journal of Engineering Manufacture, vol. 13, pp 225-246, ISSN 2041-2975, 1999.

[Moltchanov 2009] Moltchanov, B., Knappmeyer, M., Fuchs, O. and E. Paschetta, "Context management and reasoning for adaptive service provisioning", International Conference on Ultra Modern Telecommunications & Workshops, St. Petersburg, 2009, pp. 1-6.

[Moore 2007] P. Moore, P., Bin, H., Xiaomei, Z., Campbell, W. and Ratcliffe, M., "A Survey of Context Modelling for Pervasive Cooperative Learning", First IEEE International Symposium on Information Technologies and Applications in Education, ISITAE '07, vol. 5, no. 6, pp. 23-25, 2007.

[Nanda 2005] Nanda, J., Simpson, T. W. and Kumar, S. R. T., "A Methodology for Product Family Ontology Development using Formal Concept Analysis and Web Ontology Language", ASME Journal of Computing and Information Science in Engineering, Special Issue on Computer-Supported Collaborative Product Development, 2005.

[Neovius 2006] Neovius, M., Sere, K., Yan, L. and Satpathy, M., "A Formal Model of Context-Awareness and Context-Dependency", Fourth IEEE International Conference on Software Engineering and Formal Methods (SEFM'06), Pune, pp. 177-185, 2006.

[NIST 2010] NIST Manufacturing Portal, Manufacturing Ontologies Portal, Available online at: http://www.nist.gov/manufacturing-ontologies-portal.cfm, 2010.

[Obitko 2008] Obitko, M., Vrba, P., Mařík, V. and Radakovič, M., "Semantics in Industrial Distributed Systems", In: Proceedings of the 17th IFAC World Congress, pp. 13880–13887, 2008.

[OMG 1996] OMG (Object Management Group), "The Common Object Request Broker: Architecture and Specification, revision 2.0, 1996", available online at http://www.opengroup.org/branding/prodstds/x98or.htm.

[Qin 2007] Qin, W., Shi, Y. and Suo, Y., "Ontology-based context-aware middleware or smart spaces", in Tsinghua Science and Technology, vol. 12, no. 6, pp. 707-713, Dec. 2007.

[Rzevski 2007] Rzevski, G., Skobelev, P. and Andreev, V. (2007), "MagentaToolkit: A Set of Multi-agent Tools for Developing Adaptive Real-Time Applications", In: HoloMAS'07, LNCS, vol. 4659, pp. 303–313, 2007.

[Sandkuhl 2007] Sandkuhl, K and Billig, A., "Ontology-based artefact management in automotive electronics", International Journal of Computer Integrated Manufacturing, Vol. 20, Issue 7.

[Sattanathan 2006] Sattanathan S., Narendra N. C. and Maamar Z., "Ontologies for Specifying and reconciling Contexts of Web Services", Electronic Notes in Theoretical Computer Science, vol. 146, no 1, pp.43-57, 2006.

[Schilit 1994] Schilit, B., Adams, N. and Want, R., "Context-Aware Computing Applications", IEEE Workshop on Mobile Computing Systems and Applications, IEEE Computer Society 85-90, 1994.

[Schneider 2004] Schneider, P., Hayes, P. and Horrocks, I., "OWL Web Ontology Language Semantics and Abstract Syntax", W3C Recommendation: http://www.w3.org/TR/owl-semantics/, 2004.

[Self-Learning 2012] Self-Learning' (Reliable Self-Learning Production Systems Based on Context Aware Services) project of European Union's 7th Framework Programme, www.selflearning.eu, 2009-2012.

[Shirley 1992] Shirley, J., "Guide to Writing DCE Applications", O'Reilly & Associates Inc, ISBN 1-56592-004-X, 1992.

[SIRENA 2005] SIRENA Project - Service Infrastructure for Real time Embedded Networked Applications, EU-ITEA Programme, available online at http://www.sirena-itea.org/Sirena/Home.htm, 2002-2005.

[SOCRADES 2009] SOCRADES Project – Service Oriented Cross-Layer Infrastructure for Distributed Smart Embedded Devices, EU-IST 6th framework programme, available online at www.socrades.eu, 2006-2009.

[SODA 2008] SODA Project – Service Oriented Device and Delivery Architecture, EU-ITEA, programme, available online at http://www.soda-itea.org/Home/default.html, 2006-2008.

[Srikant 1997] Srikant, R. and R. Agrawal, *Mining generalized association rules.* Future Generation Computer Systems, 13(2-3): p. 161, 1997.

[Stecke 1983] Stecke, K., "Formulation and solution of non-linear Integer Production Planning Problems for Flexible Manufacturing Systems", Management Science, vol. 29, no. 3, pp. 273-288, 1983.

[Uddin 2010] Uddin, M. K., Soto, M. C., Martinez Lastra, J. L., "An integrated approach to mixed-model assembly line balancing and sequencing", *International Journal of Assembly Technology and Management (Assembly Automation)*, vol. 30(2) pp.164 – 172, 2010

[Uddin 2011] Uddin, M. K., Dvoryanchikova, A., Lobov, A., Martinez Lastra, J.L., "An ontology-based semantic foundation for flexible manufacturing systems", 37th Annual Conference on IEEE Industrial Electronics Society, IECON, vol., no., pp.340,345, 7-10, 2011.

[Uddin 2012] Uddin, M. K., Puttonen, J., Scholze, S., Dvoryanchikova, A., Martinez Lastra, J. L., "Ontology-based context-sensitive computing for FMS optimization", Assembly Automation, vol. 32(2), pp.163 – 174, 2012

[UPnP] The UPnP Forum, available at http://www.upnp.org.

[Verstichel 2008] Verstichel, S., Strobbe, M., Simoens, P., Turck, F. D., Dhoedt, B. and Demeester, P., "Distributed Reasoning for Context-Aware Services through Design of an OWL Meta-Model", Fourth International Conference on Autonomic and Autonomous Systems (ICAS'08), Gosier, 2008, pp. 70-75, 2008.

[Vyatkin 2005] Vyatkin, V., Christensen, J. and Martinez Lastra, J. L., "OOONEIDA: An Open, Object-Oriented Knowledge Economy for Intelligent Industrial Automation", IEEE Transactions on Industrial Informatics, vol. 1, no. 1, pp. 4–17, 2005.

[Wang 2004] Wang, X. H., "Ontology based context modelling and reasoning using

OWL", Orlando, FL, USA: IEEE Comput. Soc., 2004.

[W3C 2004] W3C Recommendation, 2004. Available online at: https://www.w3.org/TR/owl-guide/

[Zhou 2004] Zhou, J., Dieng-Kuntz, R., "Manufacturing ontology analysis and design: towards excellent manufacturing", 2nd IEEE International Conference on Industrial Informatics (INDIN), pp.39-45. 2004.

[Zuniga 2001] Zuniga, G. L., "Ontology: its transformation from philosophy to information systems", Ogunquit, MN, USA: ACM, 2001.

[Zuraini 2010] Zuraini, Z. and Keiichi, N., "Generic Context Ontology Modelling A Review and Framework", 2nd International Conference on Computer Technology and Development ICCTD 2010.

[Zakwan 2010] Zakwan, J., Xiaodong, L., Sally, S., "CANDEL: Product Line Based Dynamic Context Management for Pervasive Applications", International Conference on Complex, Intelligent and Software Intensive Systems, 2010.

Publications

Publication I

An integrated approach to mixed-model assembly line balancing and sequencing

Mohammad Kamal Uddin, Marian Cavia Soto, Jose L. Martinez Lastra, (2010) "An integrated approach to mixed-model assembly line balancing and sequencing", Assembly Automation, Vol. 30 Iss: 2, pp.164 - 172

DOI: http://dx.doi.org/10.1108/01445151011029808

Publication II

An ontology-based semantic foundation for flexible manufacturing systems

Uddin, M.K.; Dvoryanchikova, A.; Lobov, A.; Lastra, J.L.M., "An ontology-based semantic foundation for flexible manufacturing systems," in *IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society* , vol., no., pp.340-345, 7-10 Nov. 2011

DOI: 10.1109/IECON.2011.6119276

Publication III

Service oriented computing to Self-Learning production system

Uddin, M.K.; Dvoryanchikova, A.; Lastra, J.L.M.; Scholze, S.; Stokic, D.; Candido, G.; Barata, J., "Service oriented computing to Self-Learning production system," in *Industrial Informatics (INDIN), 2011 9th IEEE International Conference on* , vol., no., pp.212-217, 26-29 July 2011

DOI: 10.1109/INDIN.2011.6034875

Publication IV

Ontology-based context-sensitive computing for FMS optimization

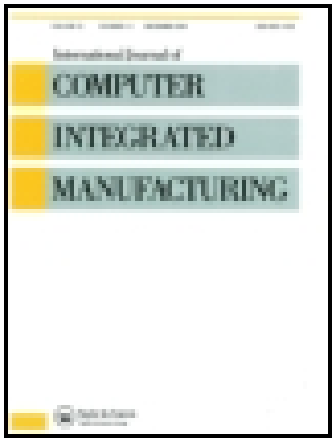
Mohammad Kamal Uddin, Juha Puttonen, Sebastian Scholze, Aleksandra Dvoryanchikova, Jose Luis Martinez Lastra, (2012) "Ontology-based context-sensitive computing for FMS optimization", Assembly Automation, Vol. 32 Iss: 2, pp.163 – 174

DOI: http://dx.doi.org/10.1108/01445151211212316

Publication V

Context-sensitive optimisation of the key performance indicators for FMS

Mohammad Kamal Uddin , Juha Puttonen , Jose Luis Martinez Lastra
International Journal of Computer Integrated Manufacturing
Vol. 28, Iss. 9, 2015



International Journal of Computer Integrated Manufacturing

Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tcim20

Context-sensitive optimisation of the key performance indicators for FMS

Mohammad Kamal Uddin[a], Juha Puttonen[a] & Jose Luis Martinez Lastra[a]

[a] Department of Production Engineering, Tampere University of Technology, FI-33101 Tampere, Finland

Published online: 02 Sep 2014.

To cite this article: Mohammad Kamal Uddin, Juha Puttonen & Jose Luis Martinez Lastra (2014): Context-sensitive optimisation of the key performance indicators for FMS, International Journal of Computer Integrated Manufacturing, DOI: 10.1080/0951192X.2014.941403

To link to this article: http://dx.doi.org/10.1080/0951192X.2014.941403

International Journal of Computer Integrated Manufacturing, 2014
http://dx.doi.org/10.1080/0951192X.2014.941403

Context-sensitive optimisation of the key performance indicators for FMS

Mohammad Kamal Uddin*, Juha Puttonen and Jose Luis Martinez Lastra

Department of Production Engineering, Tampere University of Technology, FI-33101 Tampere, Finland

(*Received 31 October 2012; accepted 23 June 2014*)

This article presents a context-sensitive optimisation approach for flexible manufacturing systems (FMSs), considering dynamic machine utilisation rate and overall equipment effectiveness (OEE) as the key performance indicators (KPIs). Run-time contextual entities are used to monitor KPIs continuously to update an ontology-based context model and subsequently convert it into business-relevant information via context management. The delivered high-level knowledge is further utilised by an optimisation support system (OSS) to infer optimal job (re)scheduling and dispatching, keeping a higher machine utilisation rate at run-time. The reference architecture is presented as add-on functionality for FMS control, where a modular development of the overall approach provides the solution generic and extendable across other domains. The key components are functionally implemented to a practical FMS use-case within service-oriented architecture -based control architecture. Test runs are performed in a simulated environment provided by the use-case control software, and the results are analysed, which indicates an improvement of the dynamic machine utilisation rate and the enhancement of the OEE.

Keywords: context; flexible manufacturing systems (FMS); optimisation; key performance indicator (KPI); service-oriented architecture (SOA); web service (WS); web ontology language (OWL)

1. Introduction

Dynamic job (re)scheduling and dispatching are becoming an essential part of modern FMS plant controls in addressing chaotic nature of the production environment. Process plans in FMS need to consider several factors at run-time, like demand fluctuations, extreme product customisations and run-time priority changes. To meet this plant-level dynamism, complex control architectures are utilised to provide an automatic response to the unscheduled/unexpected events. These run-time responses from the control systems deal with final moment change of the control parameters, which eventually influences the key performance indicators (KPIs) such as machine utilisation rate and overall equipment effectiveness (OEE). Execution of the plant-level objectives (e.g. keeping a higher machine utilisation rate at run-time) in a chaotic job processing order is hence becoming crucial. In response, modern FMS plant controls are moving towards more decentralised and adaptive control architectures, promoting the integration of different support applications for global factory optimisation (Uddin, Dvoryanchikova, Lobov, et al. 2011).

An approach for context-sensitive computing for FMS optimisation (Uddin et al. 2012) defines the state-of-the-art technologies and associated core functionalities. The system integration is addressed on top of the generic FMS control platform as a decision-support application. A context-sensitive computing defines how to extract manufacturing context at source, how to represent contextual entities formally through an ontology-based context model, how to provide a common interface for context acquisition, how to reuse and update and how to address context management to deduce high-level knowledge from the raw monitored data. The optimisation is envisioned by means of a support application that consumes critical change in contexts, adapts content and suggests optimal conditions. A partial implementation (development of lower level components) is reported to a practical FMS use-case within service-oriented architecture (SOA)-based control architecture. The lower level implementation provides the functional requirements of extracting standardised monitoring data from use-case web services (WSs), converting those to web ontology language (OWL) instances, populating and updating of dynamic entities to an ontology-based context model.

In this manuscript, detail features and functionalities of the overall approach of context-sensitive optimisation for FMS are reported, focusing on the run-time device-level KPI optimisation. The reference architecture is presented introducing higher level components for context management and their functionalities. An optimisation support system (OSS) is addressed, where an optimisation algorithm consumes contextual changes specified to the KPIs and suggests optimal job (re)scheduling and dispatching, maintaining a higher machine utilisation rate at run-time. A graphical user interface (GUI) is developed where the optimisation proposals are suggested for the operator's feedback. Several SWRL (semantic web rule language) rules bridged with Jess rule engine (O'Connor et al.

*Corresponding author. Email: mohammad.uddin@tut.fi

2005) provide the reasoned knowledge (updated periodically by GUI component) about process and devices to the GUI for operator's decision support. The GUI components are also responsible to validate the operator's action (accept/reject). Context-sensitive optimisation of the KPIs is functionally implemented to the mentioned FMS use-case, and the higher level implementation details are reported.

The contribution of this manuscript is threefold. First, a brief literature review is presented, identifying the context-sensitive computing in manufacturing and the needs for KPI optimisation in a dynamic environment of FMS (Section 2).

Second, presenting the reference architecture, the overall approach is discussed from the implementation point of view of KPI optimisation, within four interoperable modules. The higher level components and their functionalities for context management and context-sensitive optimisation, including optimisation algorithm, are introduced. User interface is also presented as a platform for the operator to interact with optimal conditions while decision-making (Section 3).

Finally, the control principles of the mentioned use-case are presented in brief, and the overall implementation of the proposed approach is reported (Section 4). Several modular tests (JUnit tests) are performed to check the component's functionality and interoperability within the modules. The integrated system is tested by formulating five test scenarios in a simulated environment, which is provided by the use-case control software. The results are analysed focusing on dynamic machine utilisation rate in a chaotic job processing order (Section 5). The conclusions are drawn, and future works are outlined (Section 6). Protégé 3.4.4 (Protégé 2009), Jena API and Java IDE (Eclipse) tools are used for implementation.

2. Literature review

In manufacturing, the core functionalities of context-sensitive computing is provided by a holistic context model that considers the context of processes, equipment, products and the utilisation of (*a priori*) knowledge for planning of activities. However, there are several associated challenges of dynamic context modelling, context capturing and context processing to fulfil real-time application's needs (Bettini et al. 2010). Plant floor-level data are variable (often continuously variable) as is the quality captured from varied process and diverse range of sensor types; context modelling approach must therefore inherently support quality and richness indication. Contextual information may suffer from incompleteness and ambiguity as well. Context modelling and processing must incorporate the capability to handle these issues (Huang and Webster 2004).

Most common approaches for context modelling are key-value models, mark-up scheme models, graphical models such as UML, object-oriented model, logic-based models and ontology-based models (Moore et al. 2007). Comparison of different context modelling techniques is reported by some researchers (Bettini et al. 2010). However, considering the level of formalism, distributed composition, applicability to existing environment and validation, the present research on context modelling is mostly focused on ontologies (Sattanathan, Narendra, and Maamar 2006). The recent advancement of ontologies in manufacturing (Lin et al. 2011) offers the creation of a common language for sharing manufacturing knowledge among designers, design tools and software applications. Formally represented manufacturing knowledge allows context-sensitive applications development for decision support (Uddin, Dvoryanchikova, Lobov, et al. 2011; Sandkuhl and Billig 2007).

FMS process planning is mainly associated with pre-release and post-release decisions. Pre-release decisions (e.g. machine grouping, part type selection) include operational planning problems with pre-arrangement of jobs and tools before actual operations and addressed during FMS configuration planning. Different optimisation methods (Uddin, Soto, and Lastra 2010) based on mathematical modelling; heuristics, meta-heuristics and simulation-based approaches are in use to address pre-release problems and to reduce the computational burden (Chen et al. 2011). Post-release operations in modern FMS plants are mainly responsive decisions to sudden changes (e.g. reactive job scheduling due to priority rule change) and to unexpected events (e.g. machine failure). At the same time, optimal decision-making in response to those changes is important as it affects the KPIs directly (Efthymiou et al. 2011). Run-time optimisation of the KPIs (e.g. higher machine utilisation) is hence becoming more challenging, yet necessary, because a high investment is associated to the processing devices. Assessment of competiveness for the manufacturing companies is also dependent on KPIs. Each level of a manufacturing company, from the higher enterprise level to the lower device level, has its own individual KPIs, and correlating these different KPIs is a potential challenge (Juarez and Landryova 2012).

In response, manufacturing controls and associated research are demanding for adaptive and decentralised control architectures having dynamic decision-making capability to the changing scenarios (Orozco and Martinez Lastra 2006). These complex controls also support the integration of different application-specific expert systems or decision-support systems (DSS) that participate and help the decision-making process of the human in an optimal way. Several intelligent DSS, such as group DSS, distributed DSS and decision support, based on knowledge discovery or natural language and their functionalities are reported in literature (Faguo et al. 2008). However, they are mostly addressed to the pervasive computing environment.

With the advancement of semantic web technologies, knowledge-intensive and context-sensitive applications supports are extensively researched within the ubiquitous

and pervasive computing environment (Stokic, Scholze, and Barata 2011). However, the research on context awareness is growing in different areas of manufacturing and relatively new to the production environment like FMS (Chen, Liu, and Wang 2008). The potential advantages of bridging context-sensitive computing to the chaotic operation environment of FMS are mainly to facilitate manufacturing knowledge management among the design tools and contextual knowledge exchange in a complex, adaptive environment.

3. Context-sensitive optimisation for FMS

Context-sensitive optimisation for FMS refers to a generic decision-support application, integrated on top of the existing plant control platform. The aim is to optimise device-level KPIs dynamically, taking run-time contextual changes into account. KPIs considered for optimisation are higher machine utilisation rate and the OEE. The reference architecture is depicted in Figure 1, consisting of four core modules: (1) Interface to plant control platform and data access layers for context extraction and update, (2) Context management for processing of the extracted context to deduce high-level knowledge contexts, (3) context-sensitive optimisation as an OSS module and (4) A GUI to integrate the user in the loop and to implement the optimal decision.

User-oriented integration provides a mean for plant managers/operators to improve the operations by better understanding and assessing the impact of suggested optimal conditions. The key features and functionalities include:

(1) Interface to plant floor control and data access layer: The existing data access layers are considered as the sources for context extraction. Appropriate level of interfacing between model-based monitoring (OWL ontology) and data access layers ensures the required level of connectivity and interoperability. This allows the extraction of contextual entities, conversion of those entities to OWL instances and populating and updating of context model for higher level processing. Information about control parameters, process, product and resources is extracted from the existing data servers, service database, plant SCADA in terms of text files, measurement/sensor data, NC configuration files, XML files and WS description language (WSDL) files depending on specific plant-level architecture.

(2) Context management: Context model, context interpretation, context repository and context brokering are the core components of the context management layer.

- *Context model:* An ontology-based (OWL) context model functions as a primary data model for context extraction and update. The associated contextual entities are run-time process preferences of products, device skills and physical capabilities of the equipment. A semantic model (ontology) provides the formal representation flexible enough to support common modelling of context in a structured way, as well as domain-specific extension to the model.
- *Context interpreter:* Context interpretation utilises the populated context model (with raw monitored data) to process meaningful knowledge context. The process utilises a model repository, context identification

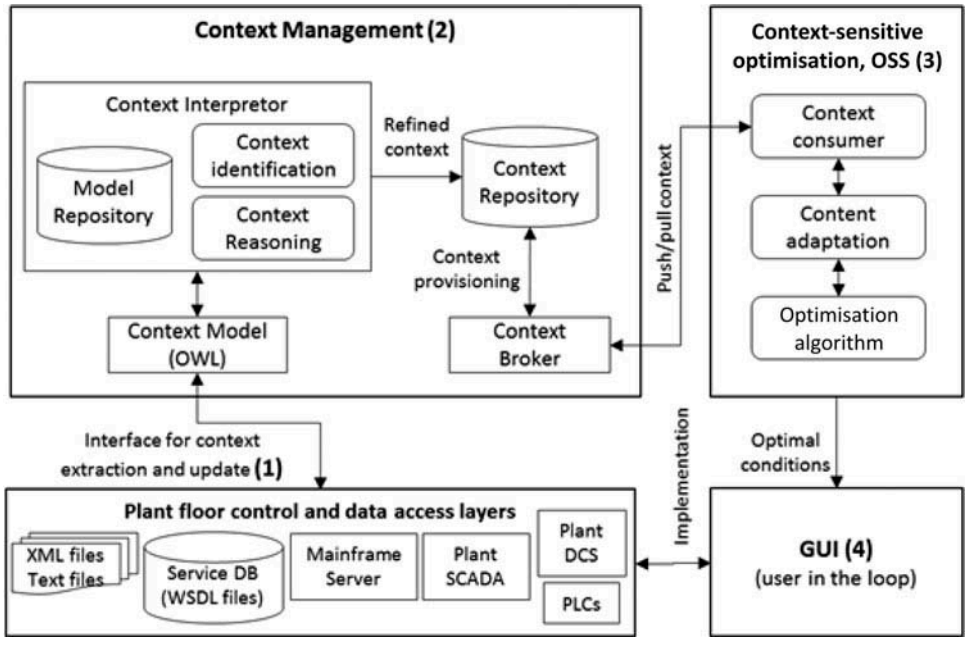


Figure 1. Reference architecture of the proposed approach.

module and rule-based reasoning to provide the required functionalities. The model repository contains ontology model for plant-specific resources, production processes and products. It allows processing of SPARQL queries (SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query/) at run-time. Context identification module maps the monitored information with the model repository to identify support application-specific instances (e.g. KPI-relevant instances) and creating a refined context model. Context reasoning is further applied to deduce a high-level, implicit context from low-level, explicit context and to check consistency and reliability. Domain-specific rule-based reasoning (SWRL rules) allows inferring implicit knowledge about the status of running system that are transferred to the GUI for as a mean for user's decision support. The refined context model is stored in repository as reference context and updated periodically.

- *Context repository:* Context repository allows update and storage of extracted/processed contextual information for later retrieval. A context repository is addressed by using relational database (RDB) and Jena framework. Java-based tool Jena (Dickinson 2009) exposes the data contained in RDBs as virtual resource description framework (RDF) graphs that can be navigated or queried as a SPARQL endpoint. The tool provides a general-purpose mechanism for mapping tables and columns of RDBs to the classes and properties of an ontology. Jena is designed to require minimal configuration to quickly expose a RDB to RDF knowledgebase utilising semantic database (SDB). SDB is a Jena component for RDF storage and query specifically to support SPARQL, which is integrated on top of an SQL database. An SDB store can be accessed and managed with the provided command-line scripts and via the Jena API.
- *Context broker:* A context broker disseminates contextual knowledge to the support application relevant to the KPIs. The broker utilises context provisioning through SPARQL queries to carry pertinent data and delivers those to the support applications. Run-time contextual knowledge can be provided proactively (proactive context provisioning, push) or based on the request from support application (on demand context provisioning, pull), specified to the application's need.

(3) Context-sensitive optimisation: The functionalities of this module are realised by an OSS, having the interoperability with context management and GUI components. An optimisation algorithm remains in the core of OSS. The flow diagram of the algorithm is depicted in Figure 2. Optimisation cycle initially starts from context broker in context management layer. Broker periodically analyses the refined context model from the repository via context provisioning and push current context to the optimiser (OSS). Context consumption and content adaptation components read the context to map predefined instances of KPI with the current context and extract the properties.

Context similarity measurement compares current context with the reference context to identify how similar they are, a real number between 0 and 1. A text-based similarity measurement (Ahn et al. 2005) is widely adopted; however there are associated limitations in the domain of search engines, collaborative filtering and clustering. For instance, two same terms can have different meaning and vice versa. A hierarchical domain structure model (Ganesan, Garcia-Molina, and Widom 2003) overcomes such limitations, and this model is exploited in implementing the FMS use-case. The following algorithm compares the similarity between two sets of same KPI instances from refined reference context model and current context model.

Input: Two sets of KPI instances to be compared
Set R [r_1, r_2, $r_3 \ldots r_n$] and set C [c_1, c_2, $c_3 \ldots c_n$]
Output: Similarity measure s
Start
s=0
If set R is not null
For each element r_i in set R
Compute its similarity with each element in set C
Select max value as the similarity measure S_i
Add s_i to s
Return s=s/(the number of set R elements)
Else
Return s=1
End

e.g. Compute Sim (r, c) = [2 × depth{LCA(r,c)}]/[hierarchy tree depth(r) + hierarchy tree depth(c)]

Similarity values can be asymmetric in using this model. However, only maximum result (i.e. similarity value 1) is considered in determining similar contextual situation.

When a contextual change is identified (context similarity value between 0 and 1) at KPI instance level, the machine utilisation rate is computed for that particular context situation (currently selected production rules) and for the alternative rules. Examples of production rules in FMS for job scheduling and dispatching are FIFO (first in first out), FOFO (first off first on), ORDER PRIORITY (job processing as per the priority order) and LOAD TIME (job processing as per lowest/highest load time).

Machine utilisation in manufacturing is usually computed using the ratio of real machining time and planned machining time. The algorithm considers the NC program time (program execution time) as the real machining

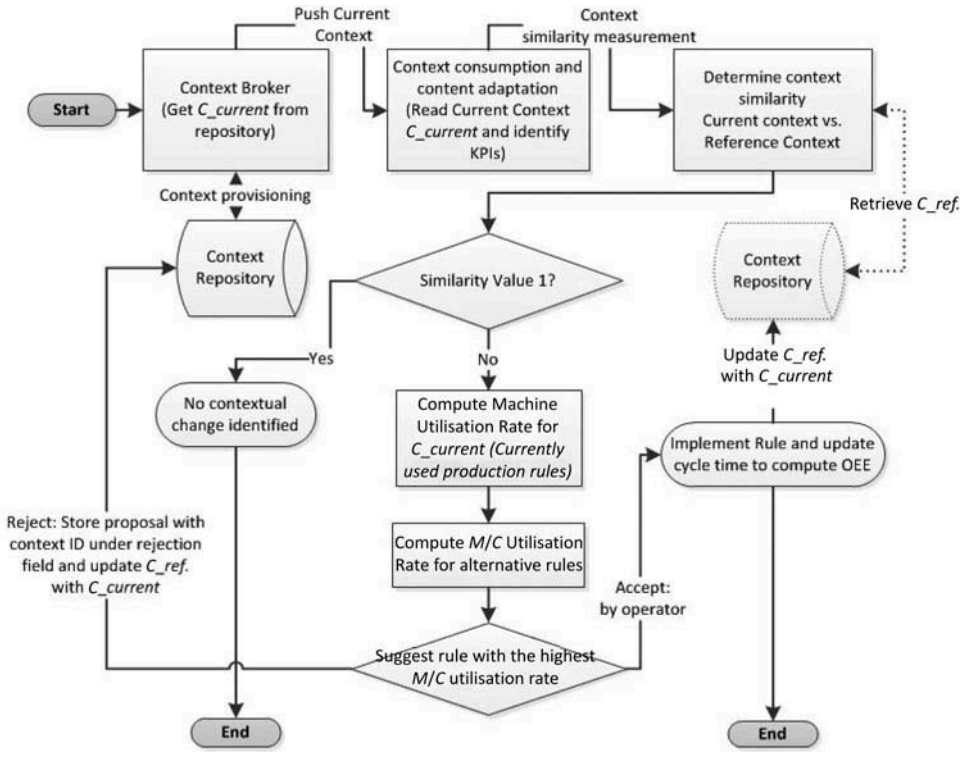


Figure 2. Flow diagram of the optimisation algorithm.

times. Planned machining time is either 24 hours or planned time from the factory calendar, for instance 24 hours planned maintenance.

The rule, having the highest machine utilisation, is suggested to a separate GUI for user's feedback (accept/reject proposed rule based on current context). In either of the actions, reference context in the repository is updated with the current context.

(4) User interface: A GUI enables the OSS component to interact with a human operator for final decision-making. The GUI presents optimisation proposals to the operator for acceptance or rejection. In addition, the operator is able to modify the proposals before accepting them. The proposal is implemented to the system once the operator accepts, and the reference context is updated as the current context. The result of operator's action is also stored under accepted proposal field with associated context ID in repository. The GUI just stores the result and associated context ID in the rejected proposal field if the operator rejects the optimal proposal. Optimal rule is implementable by the GUI component to the system addressing a dynamic reschedule of the jobs accordingly. Those components also compute the OEE, multiplying the availability, performance and quality of production. GUI takes the inputs for available times, scheduled production and good and bad parts produced within a specified shift time from the user. However, the cycle time as defined by the use-case (produced units per time unit) is updated to the GUI automatically from production log for mentioned shift time while computing the availability.

4. A FMS use-case implementation

The overall approach of context-sensitive optimisation of the KPIs is implemented to a real FMS use-case. The use-case produces different parts for industrial robots, hydraulic components and aircraft parts for automotive industries utilising automated pallet-based machining centres. Automation system of the use-case is based on rail-guided vehicles (stacker crane) or industrial robots and combination of them. Buffers in front of processing stages provide smooth operation by compensating the characteristic changes of job processing times, as there is no system characteristic processing cycle (tact) in the system. This is due to the parallel processing of production orders (chaotic processing order).

Prerequisite for FMS operation is that the set-ups of all manufactured items are ready in the system. There is no need for set-up changes in production machines or transportation equipment between the processing of jobs with different items. This is normally achieved by using a standard part carrier (pallet, zero-point fixture) and combining items and carriers with sets of zero-point information. The process chain (Uddin et al. 2012) of the use-case addresses automatic

transfer of pallets within the system, where pallets are utilised as the job-carrying entity to loading stations and machining cells. Stacker cranes are used for pallet transportation.

In the following sub-sections, principles of the use-case control application are briefly presented. Implementation objectives specified to the use-case and overall implementation details are also reported.

4.1. Control application software

Use-case control system architecture is based on SOA principles, where all the production-relevant entities offer WSs to a Microsoft.Net-based control platform. A control application software (Figure 3) runs the FMS in real-time invoking data from available services (WSDL files). The application contains a set of master data for product manufacturing and enables the operations to run in a simulated environment creating the process devices (one stacker crane for pallet handling, two loading stations, two machining stations and pallet storage). The application software is also utilised for cross-platform communication enabling different client applications support (e.g. proposed CSOSS) based on WS interfaces. The service configuration file contains the description of available interfaces and URLs to access them. Different published WSs announce the list of possible loading/unloading jobs, pallet queue, device status and other associated process parameters that can be invoked at run-time.

Once the services are initiated using the 'Start Service' button, the service components run in the background. Application UI (Figure 3) generates basic manufacturing data as the number of pallets to be added and the number of orders and rounds per order to be manufactured. The application stores all production and pallet events while running in different persistence mode. A configured SQL server saves the production log and states persistence.

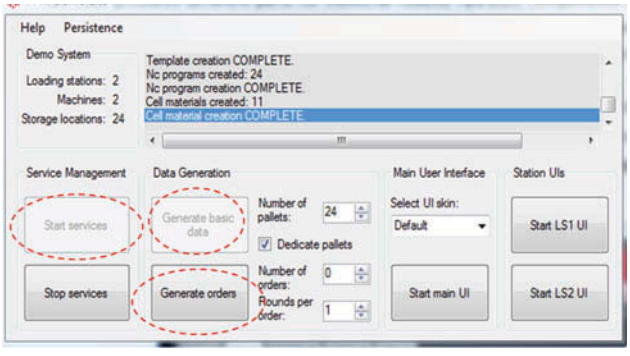


Figure 3. Use-case control application software UI.

4.2. Implementation objectives

The controller utilises a reactive scheduler for run-time job (clamped into the pallets) scheduling and dispatching. An event condition action algorithm is used in this regard. An event occurs when a loading station is free to take a pallet. The condition follows the production rule set by the operator. The action refers to creating a schedule/reschedule for pallet dispatching based on the selected rule. Reactive scheduling refers mainly to the variation of production rules (e.g. pallet sorting based on load time/order priority) dynamically to address the changing scenarios at plant floor level (e.g. addition of new jobs in the running system, shift change, unscheduled events) or at enterprise level (e.g. priority order change, due date change). Due to this rule change and associated pallet rescheduling, the machine utilisation rate is largely affected; unit production per time unit decreases (cycle time) and therefore the overall OEE is decreased.

Implementation of context-sensitive optimisation is intended to improve the dynamic machine utilisation rate by evaluating the job dispatching rules at each loading station (Figure 4). Contexts are extracted from the WSs and managed through the identification of a refined context model with inferred knowledge about process and device status. Any changes in context, relevant to the KPI (instances of a refined context model), are identified for the scenarios like the operator has changed the loading station rule, new orders are created, new pallets are added to the system, production priority value has changed, status of the machining cells has changed and some job has finished processing. The inferred knowledge from the refined context model is delivered to the GUI and updated based on context change for decision support.

The OSS computes the machine utilisation rate for the extracted current context (e.g. current rule, pallet queue, loading time, machining times, device status). Then, it computes the utilisation rate for alternative rules taking the same queuing pallets into account. If a higher utilisation rate is found, it (re)schedules the pallet dispatching

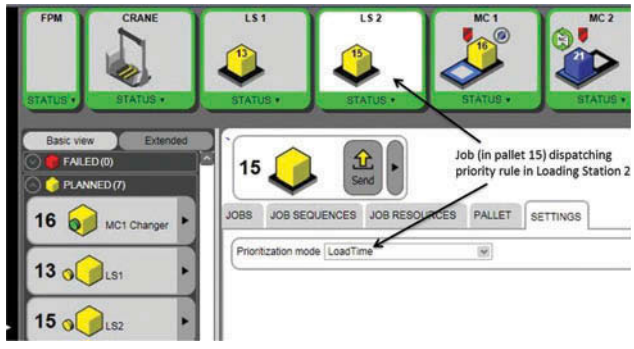


Figure 4. The main UI of the control software: loading station rule change to address reactive scheduling.

order and suggests associated rule to the GUI. In this manner, the integrated system periodically suggests the user to select a production rule and reschedule the jobs accordingly (if accepted), which have the higher machine utilisation rate for that particular context situation.

4.3. Implementation details

The overall implementation is done as a Maven-managed Java project. The project is split into two Maven modules: *cso-fms-core*, which contains the core interfaces, and classes that are envisaged to be generic for potential FMS use-cases, and *cso-fms-usecase1*, which extends the core module to account for the implemented use-case control system. The implementation of the program logic and the GUI for this specific use-case is interpreted as usecase1.

4.3.1. Context extraction interface

The interface between the WS-based SOA control platform (Uddin, Dvoryanchikova, Lastra, et al. 2011) and context model (OWL ontology) is done by creating simple Java applications that invoke the WSs from the service database to monitor the status of the production system. WSs invocation is implemented through dedicated client applications applying client stub code from WSDL description. A client application uses the stub code to invoke the corresponding WS as well as to publish the extracted information. Apache Axis2 (http://axis.apache.org/axis2/java/core/) includes command-line application, wsdl2java, for generating client stub code from WS, WSDL description. As command-line parameters, the application takes the WSDL file location, the name of the WSDL port type for which to create client code and the package name to use for the generated Java classes. The code is copied into an IDE (Eclipse) within a Java project. A Java program is written that uses client stub code to retrieve data from the WSs and populates and updates the context model with OWL instances.

4.3.2. Context extraction and modelling

A generic ontology-based context model, modelled with Protégé, is created which includes the main concepts and properties associated to FMS. Then, the model is extended for the specific use-case. Extension is achieved by creating a new OWL document which imports/reuses the generic context model and defines domain-specific concepts and properties extending those in the generic one. It may be unnecessary and computationally complex to extend the generic model to cover all aspects of the use-case that are visible through the external interfaces. Therefore, the extension is done in accordance to the production order taxonomy, and the relevant services are invoked to extract contextual entities. The invoked services are manufacturing cell service, loading station service, machine pallet service, manufacturing process service, manufacturing template repository service and NC program library service. Figure 5 illustrates the class diagram of the context model, including the mentioned concepts and associated attributes.

4.3.3. Context model population and update

To extract the contextual entities from the running services and to populate and to update the context model from the SOA platform, the following Java classes are implemented: (1) *UseCaseAnalyser* – invokes the running services and creates object model storing the monitoring data. It uses the client code generated from WSDL files to invoke the WSs and stores the retrieved data in a new *LoadingStationTasksMonitoringData* object, (2) *UseCaseMonitoringData* – provides an object

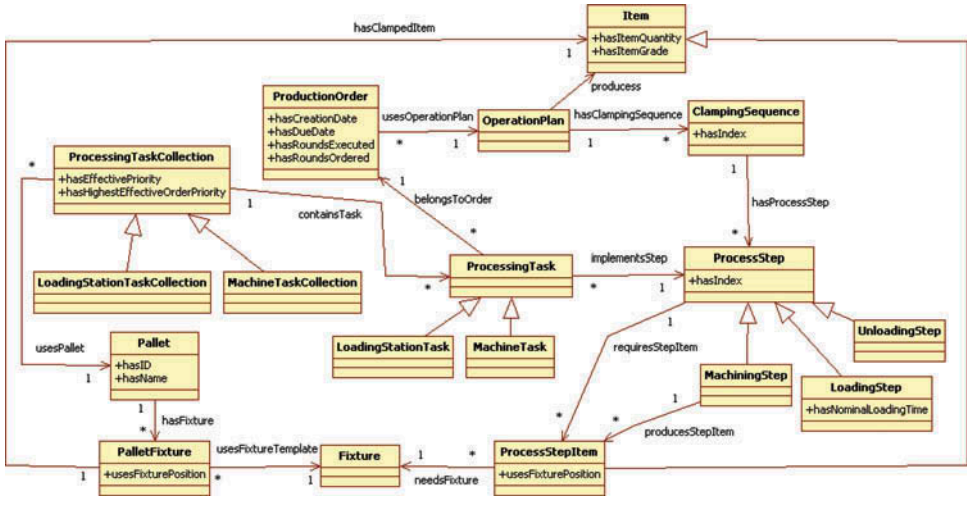


Figure 5. A class diagram of the context model.

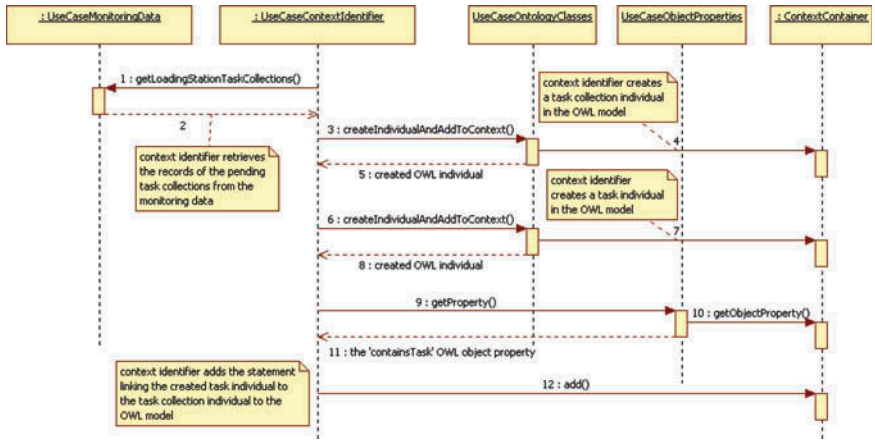


Figure 6. A sequence diagram defining the context extraction and update for the use-case.

model for extracted data on the control application software and (3) *UseCaseContextIdentifier* – analyses the obtained monitoring data and creates corresponding OWL individuals to populate the context model. Figure 6 contains a sequence diagram illustrating the process in which the context identifier projects the monitoring data of loading station tasks to the corresponding OWL individuals in the context model.

The sequence diagram is simplified and shows only the main steps of creating the individuals corresponding to the task collections and tasks as well as linking them through property assertions. The other concepts, such as production orders and operation plans, are projected to the context model using the similar process. *Junit* (JUnit.org Resources for Test Driven Development, http://www.junit.org/) tests are performed in *eclipse* environment, which provided the accurate behaviour of the *UseCaseAnalyzer* and *UseCaseContextIdentifier* class.

4.3.4. Context management

Context management is implemented by the creation of a refined context model which identifies the KPI instances (Figure 7) for computing the optimal machine utilisation rates. Rule-based reasoning is applied to infer high-level knowledge contexts, and the refined model is stored in a repository. The current implementation uses the working

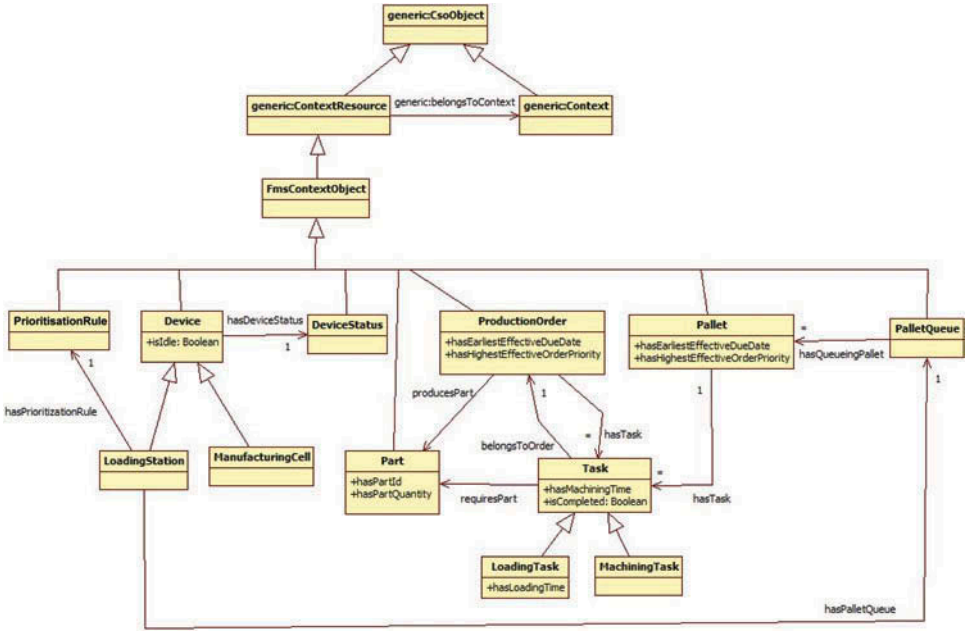


Figure 7. The class hierarchy of the refined context model with identified KPI instances for machine utilisation rate computation (contains only a few classes).

Table 1. Example of SWRL rules processed with Jess rule engine.

Sample rules	SWRL syntax
Pallets with highest production order priority	PalletPriority (?p)^hasPriorityValue(?p,?x) →Query:max(?x)
Pallets with lowest production order priority	PalletPriority (?p)^hasPriorityValue(?p,?x) →Query:min(?x)
Total machining time for the queuing pallets in LS (sum of NC programs)	JobsInQueuingPallets(?p) ^ncProgramTick (?p,?x)→Query:sum(?y)
Machining station status	MachineStation(?p) ^hasPallet(?p,true) →Busy(?p)
Loading station status	LoadingStation(?p) ^hasPallet(?p,false) →Free(?p)

memory as the context repository. Few examples of applied rules are: (1) Pallets with highest production order priority, (2) Pallets with lowest production order priority, (3) Total machining time for queuing pallets, (4) Loading station's status and (5) Machining station status.

Rule-based reasoning (inference of new knowledge by SWRL rules and Jess rule engine) examples are presented in Table 1. The GUI components also utilise this inferred knowledge to provide explicit process and device status in the OSS GUI for operator's decision support.

The refined context model is further utilised (context provisioning) by the optimiser through the context broker. The broker processes SPARQL queries to extract the knowledge from context repository and push it to the optimiser (support application). Example of run-time query to identify pallet priority queue is presented in Table 2:

4.3.5. *Context-sensitive optimisation: overall controller cycle*

The context-sensitive optimisation (OSS) analyses the control system status periodically to provide optimal proposals to the GUI. The proposals that the OSS makes entail changing the loading station prioritisation rules to either LoadTime or OrderPriority. The OSS functionalities are tailored and simplified to meet the requirements of the use-case, where the OSS consists of two main components, which are manifested as the Java classes *OptimizerGui* and *ControllerLoop*. The main control is given to the *ControllerLoop* class, which periodically analyses the use-case system status (from the refined context model), informs the GUI of the current context model and initiates the computation of a new optimisation proposal. The cycle length is set to 15 seconds. However, if the system determines that a new optimisation is in order, the next scan cycle is not started until the operator has either accepted or rejected the optimisation proposal. Figure 8 shows a simplified flow chart of the main control cycle, and Figure 9 shows a sequence diagram of the corresponding Java code execution.

4.3.6. *Graphical user interface*

The GUI is developed using the Java Swing framework. While the work has been mainly performed in the Eclipse

Table 2. Example SPARQL query to identify the priorities of the queuing pallets.

SPARQL query	Query syntax

Query to identify the run-time priority of the queuing pallets waiting to be processed	PREFIX uc1: <http://www.cso-fms.fi/ontologies/cso-fms/usecase1-refined-context.owl#>
	PREFIX gen: <http://www.cso-fms.fi/ontologies/cso-context.owl#>
	SELECT ?palletName ?minPriority
	WHERE {
	?pallet uc1:hasName ?palletName .
	?pallet uc1:hasTask ?task .
	?task uc1:belongsToOrder ?order .
	?order uc1:hasPriority ?minPriority .
	FILTER (
	NOT EXISTS {
	?pallet2 uc1:hasTask ?task2 .
	?task2 uc1:belongsToOrder ?order2 .
	?order2 uc1:hasPriority ?priority .
	FILTER (?priority > ?minPriority)
	})
	}
	ORDER BY DESC (?priority)

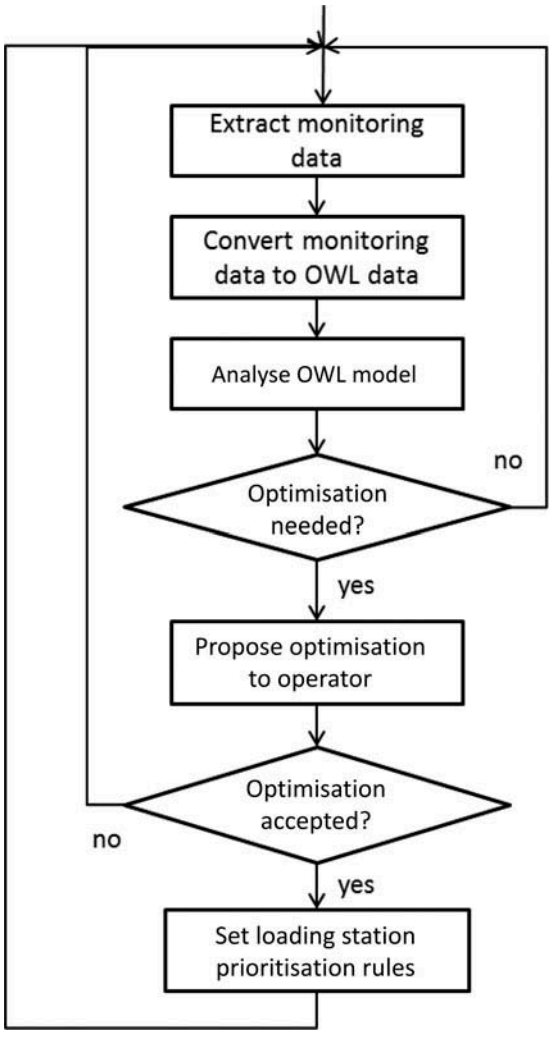


Figure 8. Each OSS control loop cycle lasts 15 seconds, or until the operator has accepted or rejected the potential optimisation proposal.

IDE, the project should be supported by alternative mainstream development environments.

The implementation of the method in Optimizer*Gui* appends the optimisation proposal in the history table. Once the operator has accepted or rejected the proposal, potentially after modifying it, the GUI executes the proposal in the system and returns the Boolean value *true*. The method returns the value *false* only when the user accepts the proposal, but executing the decision in the system fails.

The GUI consists of the main window for providing optimisation proposals comparing the machine utilisation rate for a particular contextual situation and an auxiliary dialog for computing the OEE value. The main GUI window (Figure 10) consists of two panels. The left panel shows the current status of control system (inferred knowledge from the refined context model) and the right panel shows optimisation suggestions. The right panel also allows modifying the optimisation suggestions by selecting a different prioritisation rule. Finally, the right panel allows the operator to accept or reject the proposal.

Current Process Context View (Left GUI Panel): The current process context view shows the current loading station and manufacturing cell statuses in tabular form. Devices that contain a pallet are indicated as busy. In addition, loading station statuses also include the prioritisation rule currently used in selecting loading jobs. For example, as it shows in the left panel of Figure 10, loading station 1 contains a pallet, the other devices do not contain pallets and both loading stations use the *LoadTime* prioritisation rule. The centre part of the left panel shows the current value of the parameter that the OSS aims to maximise, the machine utilisation rate. Finally, the bottom part of the panel shows the current loading station pallet queue

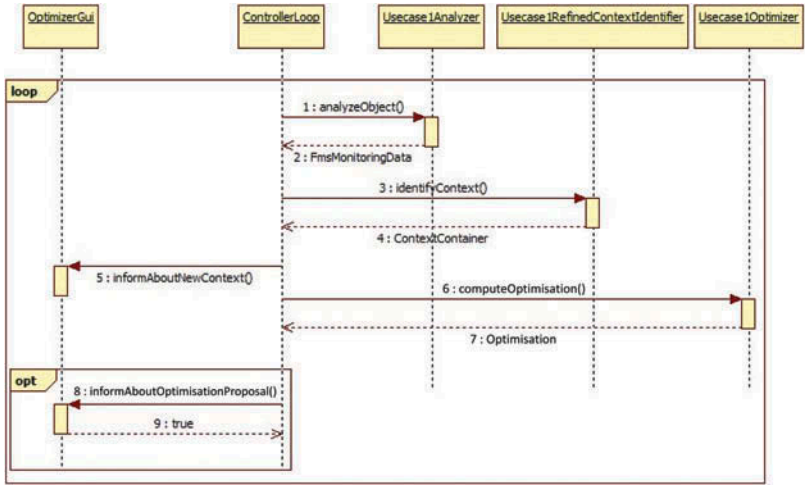


Figure 9. When the OSS is activated, it repeats the control cycle at 15-second intervals.

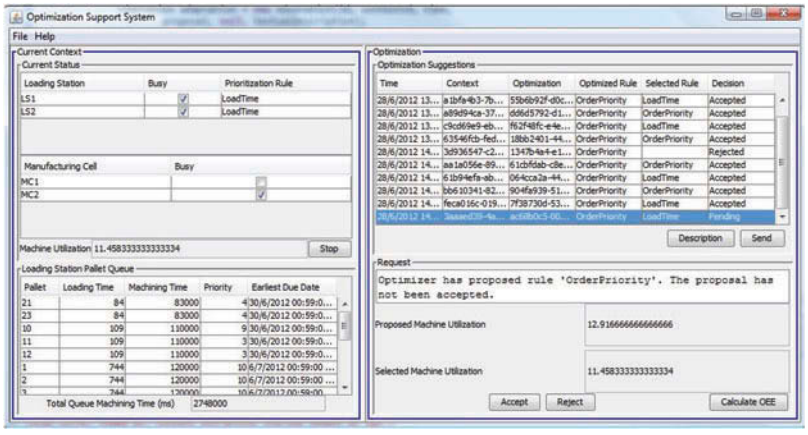


Figure 10. The main GUI window consists of two parts, current process context and optimisation suggestions.

sorted according to the current prioritisation rule. The pallet queue is used in computing the machine utilisation rate as indicated in Equation (1).

The machine utilisation depends on the machining times of the pallets in the queue.

$$U = \frac{M_1 + M_2 + M_3 \ldots\ldots + M_n}{S} \times 100 \qquad (1)$$

where U is the machine utilisation rate, M_n is the machining time of the nth pallet in the queue, and S is the shift time, which is set to eight hours.

Optimisation Suggestions View (Right GUI Panel): The right panel shows all optimisation suggestions in a table. If the last suggestion is in the *Pending* state, it can be modified by specifying a different prioritisation rule in the fifth table column, which is initially empty. If the operator does not specify any value, the system will use the proposed prioritisation rule upon optimisation acceptance. Finally, operator has the option to either accept or reject a pending optimisation proposal. In Figure 10, the right panel shows a state where the last optimisation proposal is pending for operator decision. To execute an optimisation decision in the control system, *OptimizerGui* invokes the *SetPrioritization* mode operation of the correct loading station. To invoke the operation, *OptimizerGui* uses the WS client code generated with the Axis2 WS stack and embedded into the *UseCase* module. The description of currently selected optimisation can be viewed in the lower part of the panel. In particular, the panel shows the resulting machine utilisation rates for the proposed prioritisation rule and the rule selected by the operator.

Calculate OEE Dialog: The operator can utilise the dialog for computing OEE by clicking on *Calculate OEE* button. The dialog contains a text field where the operator may enter the parameters required for computing the OEE. All values must be represented using the same unit, but otherwise the time unit is irrelevant. In addition, the dialog contains the *Update cycle time* button for retrieving the current cycle time automatically from the control system and the combo box for selecting the time unit in which the cycle time should be updated. Once the operator has entered all OEE parameters, the OEE value can be calculated by clicking on the *Calculate OEE* button (Figure 11).

5. Results and discussions

The integration of existing plant floor control (SOA based) and context management layer is a key characteristic to monitor and capture contextual entities from data access layer. Addressed Java-based WS invocation method provided the required level of details to monitor the status of production system and also periodic context model population and update considering the KPIs in scope (maximise machine utilisation rate and OEE). Context management functionality allows to maintain domain-specific ontology during the process of context-sensitive optimisation, including ontology refinement based on context identification and reasoning. The OWL-based context model provides an integrated model of knowledge and its generating and using context.

Context identification allows primary identification of high-level knowledge context from the available low-level monitoring data. Context reasoning provides precise and meaningful context out of the identified context, through a combination of ontological reasoning and rule-based reasoning. Context similarity measure utilises the hierarchical class tree defined in context model to compare similarity between reference context and current context. This hierarchical tree model requires a well-defined taxonomy of the context model; otherwise, there can be only two values computed out from the similarity measure (i.e. 1 and 0). This might occur when a primary context class has missing subclass.

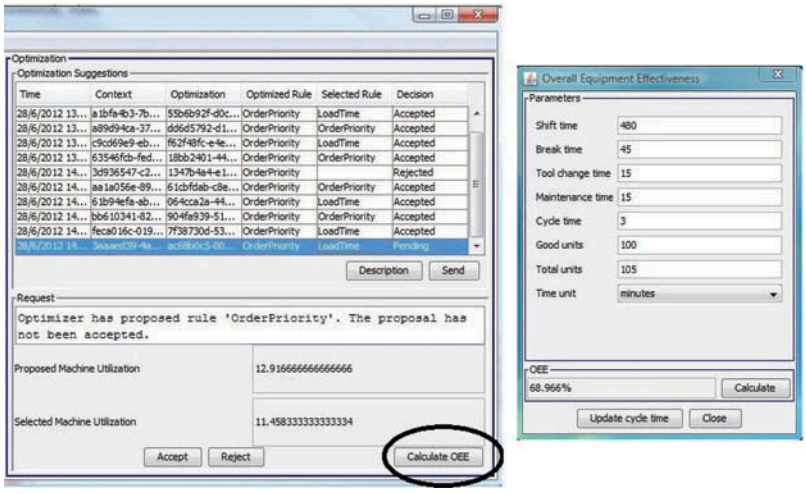


Figure 11. The OEE calculation dialog can be accessed from the main GUI window.

Domain-specific rule-based context refinement provides accurate results of similarity measurement.

Periodic contextual updates are parsed to the GUI, aiding the operator to select the optimal solution based on current device and process status. OSS analyses the machine utilisation rate on current context situation and reschedules the pallet dispatching queue accordingly. This rescheduled dispatching indicates the candidate pallets (i.e. jobs) that are possible to feed immediately and also the planned pallets that can be pushed further to achieve the calculated machine utilisation rate. Corresponding production rule is suggested to the GUI for operator acceptance. User experience is also integrated in the decision validation loop as each time operator provides a feedback to the optimal rule, the reference context is updated.

OEE is a measure for the manufactures to identify bottlenecks as far as machine utilisation rate is concern. Higher machine utilisation rate ensured by the OSS also indicates higher cycle time, meaning unit production per unit time is optimal.

To analyse the run-time behaviour of the use-case with the integrated context-sensitive optimisation, five test scenarios are performed in accordance to the modularity of the proposed approach. The use-case control application provides a simulated environment creating the process devices (Figure 4), which allows running the FMS system as a desktop computer application replicating the actual system behaviour at run-time. During the implementation phase, several JUnit tests are also performed, mainly to check the functionality and interoperability of the individual modules. Table 3 presents briefly the test scenarios and implementation results.

6. Conclusions and future works

In practice, the operating environment of FMS is highly domain specific, which in turn puts significant challenges to set unified optimal conditions. End-users of FMS utilise different run-time KPIs as maximum utilisation rate of production machines, minimise lead time of production orders, keeping the due delivery dates of production orders, minimise the tool flow in production machines or combination of them. Moreover, these optimisation objectives (KPIs) are dynamic. They are changing not only depending on the production profile but also on the process state and shift model. For example, when most of the production load is addressed for direct customer orders, it is necessary to keep the due dates as priority. When most of the production is for Kanban manufacturing (stock batches), the priority it demands is highest machine utilisation. This is the parallel rationale that the KPIs cannot be compared to any specified value, since those are mostly context dependent.

In this manuscript, the proposed context-sensitive optimisation of the KPIs demonstrates its effective integration within an existing FMS control platform, aiding the shop floor managers/operators in their dynamic decision-making process. The result of this work shows how interoperable contextual knowledge in the presented architecture can be used to address optimisation of KPIs like machine utilisation rate and OEE while considering the chaotic job processing nature of FMS plants.

Exploitation of the addressed context-sensitive optimisation enables the users to understand that the system supports to decide the optimal production rule and subsequent pallet dispatching which ensures maximum machine utilisation at that particular contextual situation.

The reference architecture and associated modular development for the use-case implementation shows that, formally modelled knowledge in contextual representation run-time context management can be productively used for run-time decision-making through support applications.

The functional implementation of the overall approach to the practical FMS use-case also provides a motivation for the

Table 3. Result analysis of the CS-OSS integration with the FMS use-case.

No	Test scenarios	Implemented actions	Implementation results
1	Test bed preparation for context extraction, modelling and update	Java clients created to invoke the interfaces provided by the use-case WSs from running services. Use-case control application is configured in the same desktop PC. Test production orders are created, and the control system is running in automatic mode for part manufacturing.	The clients invoke the WSs, convert monitoring data to OWL instances and populate the context model and update the instances periodically (15-second interval).
2	Context management	Context identification and rule-based reasoning to create a refined context model.	OWL instances are mapped onto the process model, and the KPI instances are identified, applied SWRL rules infer knowledge about process status and a refined context model is created and stored as reference context with ID and timestamp.
3	Run-time device and process status update to GUI	The inferred knowledge from the refined context model is parsed to OSS GUI for operator supervision and decision-making.	The context broker processes SPARQL queries on refined context model and the status of machining cells, loading stations, number of queuing pallets, pallet loading time, machining time, order priority and due dates are parsed to OSS GUI and updates periodically.
4	Context-sensitive optimisation	The context broker pushes the KPI instances to compute machine utilisation rate to the OSS components.	Broker processes SPARQL queries and pushes the KPI instances to OSS. The OSS executes context similarity map, identifies a change in the context and proposes optimal production rule to the OSS GUI based on the queuing pallets which indicate the highest *M*/*C* utilisation rate at that context situation.
5	Operator's action and feedback	The operator accepts/rejects/does nothing to the proposed optimal rule.	Operator accepts: optimal rule implemented to the control system, pallets are sorted accordingly for dispatching and reference context is updated with the current context. Operator rejects: Reference context is updated only with the current context. Operator does nothing: the system waits until the last proposal accepted/rejected.

further extension of this research. In the future, this research can be extended to incorporate a learning behaviour to the OSS, based on the operator's actions to the suggested optimisation proposals. Prerequisite for such learning behaviour requires the operator to be active and give feedback to each suggested optimal conditions. Otherwise, the system does not get the right data for learning and learning-based proactive decision-making. Data mining tools can be integrated to identify the contextual patterns of the operator's decisions. These patterns can be used to provide prior knowledge to the OSS. The implementation of the context repository and the online help system of the OSS GUI will be addressed with the future extension of this work.

References

Ahn, A., H. J. Lee, K. Cho, and S. J. Park. 2005. "Utilizing Knowledge Context in Virtual Collaborative Work." *Decision Support Systems* 39 (4): 563–582. doi:10.1016/j.dss.2004.03.005.

Bettini, C., O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni. 2010. "A Survey of Context Modelling and Reasoning Techniques, Pervasive and Mobile Computing." *Science Direct* 6 (2): 161–180.

Chen, Y., Z. Li, M. Khalgui, and O. Mosbahi. 2011. "Design of a Maximally Permissive Liveness-Enforcing Petri Net Supervisor for Flexible Manufacturing Systems." *IEEE Transactions on Automation Science and Engineering* 8 (2): 374–393. doi:10.1109/TASE.2010.2060332.

Chen, Z., S. Liu, and X. Wang. 2008. "Application of Context-Aware Computing in Manufacturing Execution System." IEEE International Conference on Automation and Logistics, ICAL 2008, Qingdao, September 1–3, 1969–1973. IEEE.

Dickinson, I. 2009. "Jena Ontology API." Accessed October 25, 2012. http://jena.sourceforge.net/ontology/index.html

Efthymiou, K., K. Sipsas, D. Melekos, K. Georgoulias, and G. Chryssolouris. 2011. "A Manufacturing Ontology Following Performance Indicators Approach." In *7th International Conference on Digital Enterprise Technology*, Athens, September 28–30, 586–595. ISBN 978-960-88104-2-6.

Faguo, Z., Y. Bingru, L. Li., and C. Zhuo. 2008. "Overview of the New Types of Intelligent Decision Support System." 3rd International Conference on Innovative Computing Information and Control, ICICIC '08, Dalian, June 18–20, 267. IEEE.

Ganesan, G., H. Garcia-Molina, and J. Widom. 2003. "Exploiting Hierarchical Domain Structure to Compute Similarity." *ACM Transactions on Information Systems* 21 (1): 64–93. doi:10.1145/635484.635487.

Huang, W., and D. Webster. 2004. "Enabling Context-Aware Agents to Understand Semantic Resources on the WWW and the Semantic Web." Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence, Beijing, September 20–24, 138–144. IEEE.

Juarez, E. L., and L. Landryova. 2012. "Analysis of a Model for Key Performance Indicators in an SME Assembly Line." 13th International Carpathian Control Conference (ICCC), High Tatras, May 28–31, 418–421. IEEE.

Lin, L. F., W. Y. Zhang, Y. C. Lou, C. Y. Chu, and M. Cai. 2011. "Developing Manufacturing Ontologies for Knowledge Reuse in Distributed Manufacturing Environment." *International Journal of Production Research* 49 (2): 343–359. doi:10.1080/00207540903349021.

Moore, P., H. Bin, C. W. Xiaomei Zhu, and M. Ratcliffe. 2007. "A Survey of Context Modeling for Pervasive Cooperative Learning." In *First IEEE International Symposium on Information Technologies and Applications in Education, ISITAE '07*, Kunming, November 23–25, K5-1–K5-6. IEEE.

O'Connor, M. J., H. Knublauch, S. W. Tu, B. Grossof, M. Dean, W. E. Grosso, and M. A. Musen. 2005. "Supporting Rule System Interoperability on the Semantic Web with SWRL." In *4th International Semantic Web Conference (ISWC), LNCS 3729*, Galway, 974–986. Springer Verlag. Accessed September 20, 2012. http://www.mit.edu/~bgrosof/paps/swrl-editor-iswc2005.pdf

Orozco, O., and J. L. Martinez Lastra. 2006. "Using Semantic Web Technologies to Describe Automation Objects." *International Journal of Manufacturing Research* 1 (4): 482–503. doi:10.1504/IJMR.2006.012257.

Protégé. 2009. "The Protégé Ontology Editor and Knowledge Acquisition System." Accessed October 10, 2012. http://protege.stanford.edu/

Sandkuhl, K., and A. Billig. 2007. "Ontology-Based Artefact Management in Automotive Electronics." *International Journal of Computer Integrated Manufacturing* 20 (7): 627–638. doi:10.1080/09511920701566467.

Sattanathan, S., N. C. Narendra, and Z. Maamar. 2006. "Ontologies for Specifying and Reconciling Contexts of Web Services." *Electronic Notes in Theoretical Computer Science* 146 (1): 43–57. doi:10.1016/j.entcs.2005.11.006.

Stokic, D., S. Scholze, and J. Barata. 2011. "Self-Learning Embedded Services for Integration of Complex, Flexible Production Systems." 37th Annual Conference on IEEE Industrial Electronics Society, IECON 2011, Melbourne, November 7–10, 415–420. IEEE.

Uddin, M. K., A. Dvoryanchikova, A. Lobov, and J. L. Martinez Lastra. 2011. "An Ontology-Based Semantic Foundation for Flexible Manufacturing Systems." 37th Annual conference of the IEEE Industrial Electronics Society, Melbourne, November 7–10. IEEE.

Uddin, M. K., A. Dvoryanchikova, J. L. M. Lastra, S. Scholze, D. Stokic, G. Candido, and J. Barata. 2011. "Service Oriented Computing to Self-Learning Production System." 9th IEEE International Conference on Industrial Informatics (INDIN), Caparica, July 26–29, 212–217. IEEE.

Uddin, M. K., J. Puttonen, S. Scholze, A. Dvoryanchikova, and J. L. Martinez Lastra. 2012. "Ontology-Based Context-Sensitive Computing for FMS Optimization." *Assembly Automation* 32 (2): 163–174. doi:10.1108/01445151211212316.

Uddin, M. K., J. L. Soto, and J. L. M. Lastra. 2010. "An Integrated Approach to Mixed-Model Assembly Line Balancing and Sequencing." *Assembly Automation* 30 (2): 164–172. doi:10.1108/01445151011029808.

Tampereen teknillinen yliopisto
PL 527
33101 Tampere

Tampere University of Technology
P.O.B. 527
FI-33101 Tampere, Finland

ISBN 978-952-15-3934-3
ISSN 1459-2045