PROCEEDINGS

SEKE 2009

The 21st International Conference on Software Engineering & Knowledge Engineering

Sponsored by

Knowledge Systems Institute Graduate School, USA

Technical Program

July 1-3, 2009
Hyatt Harborside Hotel, Boston, Massachusetts, USA

Organized by

Knowledge Systems Institute Graduate School

Huanjing Wang, *Western Kentucky University, USA*
Christiane Gresse von Wangenheim, *Universidate do Vale do Itaja, Brazil*
Tim Weitzel, *Bamberg University, Germany*
Victor Winter, *University of Nebraska at Omaha, USA*
Guido Wirtz, *Bamberg University, Germany*
Eric Wong, *University of Texas at Dallas, USA*
Franz Wotawa, *Technische Universitaet Graz, Austria*
Haiping Xu, *University of Massachusetts Dartmouth, USA*
Chi-Lu Yang, *Institute for Information Industry, Taiwan*
Hongji Yang, *De Montfort University, UK*
Ren-Dar Yang, *Institute for Information Industry, Taiwan*
Huiqun Yu, *East China University of Science and Technology, China*
Du Zhang, *California State University Sacramento, USA*
Jing Zhang, *Motorola Research, USA*
Min-Ling Zhang, *Hohai University, China*
Zhinan Zhou, *Samsung, USA*
Hong Zhu, *Oxford Brookes University, UK*
Xingquan Zhu, *Florida Atlantic University, USA*
Eugenio Zimeo, *University of Sannio, Italy*
Andrea Zisman, *City University, UK*

PUBLICITY CO-CHAIRS

Daniel Beimborn, *Bamberg University, Germany*
S. Masoud Sadjadi, *Florida International University, USA*

South America Liasion

Jose Carlos Maldonado, *University of Sao Paulo, Brazil*

Industry Advisory Committee

Yi Deng, *Dean, School of Computer Science, Florida International University, USA*
J. S. Ke, *Senior Fellow, Institute for Information Industry, Taiwan*
A. J. Rhem, *Senior Partner, A. J. Rhem and Associates Inc., USA*

Proceedings Cover Design

Gabriel Smith, *Knowledge Systems Institute Graduate School, USA*

Conference Secretariat

Judy Pan, *Chair, Knowledge Systems Institute Graduate School, USA*
Omasan Etuwewe, *Knowledge Systems Institute Graduate School, USA*
Chen-Cheang Huang, *Knowledge Systems Institute Graduate School, USA*
Daniel Li, *Knowledge Systems Institute Graduate School, USA*

Table of Contents

Software Engineering of Autonomic Grid Computing Systems and Applications

Requirements

Agents and Multi-Agent Systems

Interoperability and Semantic Web Technologies

The reasons for this are the multitude of mapping/matching approaches available, e.g. different matching algorithms, matching types etc. Each approach has unique requirements for mapping representation, simply because different information and structures need to be represented to express a correspondence. The design of a mapping representation which fulfils all those requirements might be too complex or could lead to a format which represents only the smallest common denominator. For example INRIA is generic but, compared to proprietary formats (e.g. FOAM) less detailed. Multiple mapping representations may be unavoidable because for different mapping scenarios, different representations of the mapping correlations are suitable. In contrast, meta-data which documents the mapping lifecycle is more uniform and for most correlation representations are available. As a result we propose that it is more beneficial to develop the concept of a flexible enrichment of existing and future ontology mapping representations in order to augment their usage, reuse and management. In particular, in an ontology based meta-layer a common vocabulary for modelling life-cycle meta-data could be established and linked to the individual formats representing mapping correlations [20]. Established mapping formats and tools don't need to be changed but available meta-data can still be stored and retrieved in a structured, documented and predictable way.

In conclusion, the remarkable efforts to support the creation of ontology mappings are just the first step. Further research is needed to develop more powerful concepts for the management, sharing and reuse of ontology mappings to even begin to support the flexible communication of a common understanding of a domain at a scale large enough to control the overall information glut [1].

ACKNOWLEDGEMENTS

This work is partially funded through the Science Foundation Ireland FAME project (award No. 08/SRC/I1408).

REFERENCES

[1] Antoniou, G., van Harmelen, F., A Semantic Web Primer (Cooperative Information Systems),The MIT Press, 2004.

[2] Gruber, T. A, Transistional Approach to Portable Ontology Specifications, Knowledge Acquisition, 5, pp. 199-220, 1993.

[3] Fensel, D., Ontologies Silver Bullet for Knowledge Management and Electronic Commerce, 2nd edition, Berlin, 2003.

[4] Corcho, O. A declarative approach to ontology translation with knowledge preservation, Volume 116 Frontiers in A.I., 2005.

[5] Euzenat, J., An API for ontology alignment, in: Proceedings of the International Semantic Web Conference (ISWC 2004), Springer, Berlin, Germany, 2004. pp. 698-712.

[6] Pepijn, R. S. V., Dean, M. J., Bench-capon, T. J. M. , Shave, M., An analysis of ontological mismatches: Het-erogeneity versus interoperability, AAAI, Spring Sympo-sium on Ontological Engineering, Stanford, USA, 1997.

[7] Bouquet, P., Ehrig, M., Euzenat, J. et al., D2.2.1 Specification of a common framework for characterizing alignment, http://www.inrialpes.fr/exmo/cooperation/kweb/heterogeneity/deli/kweb-221.pdf, 2005.

[8] O'Sullivan, D., Wade, V., Lewis, D. Understanding as We Roam, in IEEE Internet Computing, 11, (2), 2007, p26 - 33 DOI: http://doi.ieeecomputersociety.org/10.1109/MIC.2007.50

[9] Hameed, A, Preece, A., Sleeman, D., Ontology Reconciliation, Handbook of ontologies, in: Stabb S. and Suder R. (eds) International Handbooks on Information Systems, Springer Verlag, Berlin, Germany, 2004, pp 31-250.

[10] Kalfoglou, Y., Schorlemmer, M. Ontology mapping: the state of the art. in The Knowledge Engineering Review, 18(1):1–31, 2003.

[11] Euzenat et al. D2.2.3: State of the art on ontology alignment, ftp://ftp.inrialpes.fr/pub/exmo/reports/kweb-223.pdf, 2004. E04b

[12] Euzenat et al. D2.2.6: Specification of the delivery alignment format, 2006. http://www.inrialpes.fr/exmo/cooperation/kweb/heterogeneity/deli/kweb-226.pdf

[13] Shvaiko, P., Euzenat J., A Survey of Schema-based Matching Approaches, in DIT Technical Report DIT-04-87, 2004.

[14] Noy, N., Semantic Integration A Survey of Ontology-Based Approaches, in Special Issue on Semantic Integra-tion, SIGMOD Record, Volume 33, Issue 4, pages 65-70, December 2004.No04

[15] Basili, V. R., Caldiera, G., Rombach, H. D., Goal Question Metric Approach, ftp://ftp.cs.umd.edu/pub/sel/papers/ gqm.pdf, 2000.

[16] Thomas, H., O'Sullivan, D., Brennan, R.: Evaluation of Ontology Mapping Representations. In: Workshop on Matching and Meaning, Edinburgh, 2009

[17] Falconer, S., Storey, M.-A., A cognitive support framework for ontology mapping. In Proc. of the 6th Int. Semantic Web Conference, http://iswc2007.semanticweb.org/papers/113.pdf, 2007.

[18] Maedche, A., Motik, B., Silva, N. et al. MAFRA - A MApping FRAmework for Distributed Ontologies, in: Proc. of the 13th Int. Conf. on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 2002, pp. 235 - 250. Ma02c

[19] Aumüller, D. et al. Schema and Ontology Matching with COMA++. In: Proc. of the ACM SIGMOD Int. Conference on Management of Data, 2005; pp. 906-908.

[20] Thomas, H., Redmann, T., Markscheffel, B. Controlled semantic tagging, in: Shoniregun, C. A., Logvynovskiy, A.: Proc. of the International Conference on Information Society, 2007, pp. 346-352.

[21] Beigi, M., Calo, S., Verma, D., Policy Transformation Techniques in Policy-based Systems Management, in: Proc. of IEEE Policy 2004, NY, June 2004.

[22] Smith, B., Ontology and Information Systems, Lecture Text, http://ontology.buffalo.edu/ontology(PIC).pdf, 2000.

[23] Garshol, L.M., Moore, G. ISO/IEC JTC1/SC34, Information Technology, http://www.isotopicmaps.org/sam/sam-model/ (2006)

[24] Wang, P. Xu, B. Lily. Proc. of the 3rd int. workshop on Ontology Matching,08,http://dit.unitn.it/~p2p/OM2008/oaei08_paper7.pdf

[25] Ehrig, M, Sure, Y. FOAM - Framework for Ontology Alignment and Mapping; In: Ashpole, B, et al: Proceedings of the Workshop on Integrating Ontologies, volume 156, pp. 72-76. October 2005.

[26] Euzenat J., An API for ontology alignment, Int. Semantic Web Conference 2004, Springer, Berlin, Germany, 2004, pp 698-712.

[27] Caracciolo, C., et al.Results of the Ontology Alignment Evaluation Initiative 2008. Proc. of the 3rd int. workshop on Ontology Matching, 2008. http://www.dit.unitn.it/~p2p/OM-008/oaei08_paper0.pdf.

[28] Giunchiglia, F., Shvaiko, P., Semantic matching, in: proc. of the Workshop on ontologies & distributed systems,pp. 193-146, 2003.

[29] Bouquet P. et al. C-OWL: Contextualization Ontologies. In proc. of the 2nd Semantic Web Conf. 2003, pp. 164-179

APPENDIX A OVERVIEW OF EVALUATED APPLICATIONS

Application	Link
Alignment API	http://alignapi.gforge.inria.fr/
Anchor-PROMPT	http://protege.stanford.edu/plugi[illegible] prompt/prompt.html
COMA++	http://dbs.uni-leipzig.de/Research[illegible]
Context Matching Algorithm (CtxMatch)	http://dit.unitn.it/~zanobini/ downloads.html
CROSI Mapping System	http://www.aktors.org/crosi/
Falcon-AO	http://iws.seu.edu.cn/projects/m[illegible] projects.jsp
Framework for Ontology Alignment & Mapping (FOAM)	http://www.aifb.uni-karlsruhe.d[illegible] WBS/meh/foam/
Lily	http://ontomappinglab.googl[illegible] lily.htm
MAFRA	http://mafra-toolkit.sourcefo[illegible]
MapOnto	http://www.cs.toronto.edu/ semanticweb/maponto/
OntoBuilder	http://iew3.technion.ac.il/[illegible]
Ontology Mapping Tool s	http://www.wsmx.org/
Risk Minimization Ontology Mapping (RiMOM)	http://keg.cs.tsinghua.ed[illegible] MOM/

Bridging Semantic Gaps Between Stakeholders in the Production Automation Domain with Ontology Areas

Stefan Biffl, Wikan Danar Sunindyo, Thomas Moser

Institute of Software Technology and Interactive Systems, Vienna University of Technology
Favoritenstrasse 9-11/188, Vienna, Austria
{ stefan.biffl, wikan.sunindyo, thomas.moser}@tuwien.ac.at

Abstract—**Stakeholders from several domains with local terminologies have to work together to develop and operate software-intensive systems, like production automation systems. Ontologies support the translation between local terminologies via common domain concepts. Unfortunately, the ontology models can become large and complex if they include several aspects on a domain and some parts of the data model are volatile. In this paper, we propose a data modeling approach to support ontology users based on ontology building blocks, so-called "Ontology Areas" (OAs), which allow solving tasks with smaller parts of the overall ontology. We evaluate the proposed approach with use cases from the production automation domain: translation between stakeholder roles to support design-time and run-time decision making. Major result in the study context is that OAs improved the efficiency of data collection for decision making.**

I. INTRODUCTION

The integration of business processes and IT systems in homogeneous environments (i.e., consistent data formats and terminology) is supported by well-established approaches like [illegible] integration using Scheer's ARIS for CIM [21]. However, [illegible] more heterogeneous environments with a range of data [illegible] and local terminologies like the production automa[illegible] domain, typically stakeholders from several areas (e.g., [illegible] experts, software engineers and electrical engineers) [illegible] together to develop and operate software-intensive sys[illegible] A homogenization of these environments is often not [illegible], if the stakeholders come from different organiza[illegible] backgrounds or organizations change over time due to [illegible] and acquisitions. The precondition for successful se[illegible] integration is a common understanding on the relevant [illegible] the problem domain of the project.

[illegible] for a collection of common problem domain [illegible] the *Enterprise-Control System Integration*[1] (ECSI) [illegible] for developing automated interfaces between en[illegible] control systems. The objectives of ECSI are to [illegible] consistent terminology as foundation for supplier [illegible] communications, b) consistent information [illegible] consistent operations (process) models, which [illegible] clarifying application functionality and how [illegible] be used.

[illegible] standard like ECSI can only cover parts of the [illegible] without getting too complex and hard to use. [illegible] players in the production automation do[illegible] not follow this standard, which often hin[illegible] of stakeholders in projects, since transformations between stakeholder terminologies to overcome semantic gaps between the stakeholders need to be conducted by scarce experts or carefully hand-crafted.

Ontologies are flexible open-world data models for knowledge representation, which store information in machine-understandable notation [12]. Therefore, ontologies can help to bridge semantic gaps between partial data models by providing mappings between them via common domain concepts. Ontologies usually capture problem-domain-specific information which can be reused later. Due to their concurrent development ontologies need to be checked for inconsistencies to stay useful. However, ontologies in practice usually have to combine several view points and thus get large and complex, particularly, if the ontology contains volatile domain elements, such as run-time data.

In this paper, we propose a data modelling approach that helps structure ontologies with ontology building blocks, so-called *"Ontology Areas"* (OAs). An OA is a meaningful part of an ontology for a stakeholder, which helps ontology users managing a complex ontology. The combination of all needed OAs represents the overall ontology for supporting the original engineering process.

We evaluate the proposed OA approach with use cases in the production automation domain: 1. Translation between local stakeholder terminologies; 2. Provision of design context for run-time data interpretation; and 3. Run-time measurement representation for design model improvements. The use cases are based on the data model of the "Simulator for Assembly Workshops" (SAW) [15] and compare the performance of an ontology with and without OAs. The evaluation showed that OAs made the data collection in the ontology for decision support more efficient in the study context, since the OAs result in a smaller ontology for the tasks in the use cases.

The remainder of this paper is structured as follows: Section 2 summarizes related work on system integration and ontologies. Section 3 describes the industry use case and Section 4 derives research issues. Section 5 introduces the OA approach, while Section 6 evaluates the approach and discusses the results. Finally, Section 7 concludes the paper and identifies further work.

II. RELATED WORK

This section summarizes related work on system integration and ontologies for semantic integration to reconcile different views of stakeholders on system data.

A. Integration of Heterogeneous Systems

System integration is the task to combine a range of smaller systems to appear as one big system. There are several levels at which system integration could be performed [3], but there is so far no standardized integration process that explains how to integrate systems in general.

Typical integration solutions focus either on technical heterogeneity (how to connect systems that use different platforms or protocols) or on semantic heterogeneity (how to translate data in messages between systems that use different data formats or terminologies). In order to cope with technical heterogeneity on service level middleware technology [9] supports syntactical transformation between services, while the semantic heterogeneity of services can be addressed with a common data schema [13]. Limitations of these integration approaches are: 1. The need for a common data schema [13], which is hard and time-consuming to negotiate, sometimes impossible if stakeholders continue to disagree. 2. The need for integration over heterogeneous middleware technologies (with different APIs or network architecture styles) implies the development of static and therefore inflexible wrappers between each combination of middleware technologies, and thus increases the complexity of communication.

Semantic integration is defined as the solving of problems originating from the intent to share data across disparate and semantically heterogeneous data [13]. These problems include the matching of ontologies or schemas, the detection of duplicate entries, the reconciliation of inconsistencies, and the modelling of complex relations in different sources [20]. Over the last years, semantic integration became increasingly crucial to a variety of information-processing applications and has received much attention in the web, database, data-mining and AI communities [6]. One of the most important and most actively studied problems in semantic integration is establishing semantic correspondences (also called mappings) between vocabularies of different data sources [7].

B. Ontologies for Semantic Integration

An ontology is a representation vocabulary for a specific domain or subject matter, like production automation. More precisely, it is not the vocabulary as such that qualifies as an ontology, but the (domain-specific) concepts that the terms in the vocabulary are intended to capture [5]. Many authors like Goh [11] identified three main categories of semantic heterogeneities in the context of data integration that can appear: confounding conflicts (e.g., equating concepts are actually different), scaling conflicts (e.g., using different units for the same concept), and naming conflicts (e.g., synonyms).

Noy [19] identified three major dimensions of the application of ontologies for supporting semantic integration: the task of finding mappings (semi-)automatically, the declarative formal representation of these mappings, and reasoning using these mappings. There exist two major architectures for mapping discovery between ontologies: 1. It is possible to create a general upper ontology which is agreed upon by developers of different applications. Two examples for ontologies that are built specifically with the purpose of being formal top-level ontologies are the *Suggested Upper Merged Ontology* (SUMO) [18] and DOLCE [10]. 2. There are approaches comprising heuristics-based or machine learning techniques that use various characteristics of ontologies (e.g., structure, concepts, instances) to find mappings. These approaches are similar to approaches for mapping XML schemas or other structured data [4, 6]. The declarative formal representation of mappings is facilitated by the higher expressive power of ontology languages which provide the opportunity to represent mappings themselves in more expressive terms.

Uschold and Gruninger [23] identified four main categories of ontology application to provide a shared and common understanding of a domain that can be communicated between people and application systems [8]: Given the vast number of non-interoperable tools and formats, a given company or organization can benefit greatly by developing their own neutral ontology for authoring, and then developing translators from this ontology to the terminology required by the various target systems. While it is safe to assume there will not be global ontologies and formats agreed by all possible stakeholders, it is nevertheless possible to create an ontology to be used as a neutral interchange format for translating among various formats. There is a growing interest in the idea of "Ontology-Driven Software Engineering" in which an ontology of a given domain is created and used as a basis for specification and development of some software [19]. The benefits of ontology-based specification are best seen if there is a formal link between the ontology and the software. To facilitate search, an ontology is used as a structuring device for an information repository (e.g., documents, web pages, names of experts); this supports the organization and classification of repositories of information at a higher level of abstraction than is commonly used today.

As alternative approach for semantic integration of system models the infrastructure of Model-Driven Architecture (MDA) [16] provides architecture for creating models and meta-models, defining transformations between these models, and managing meta-data. Although the semantics of a model is structurally defined by its meta-model, the mechanisms to describe the semantics of the domain are rather limited compared to machine-understandable representations using, e.g., knowledge representation languages like RDF[2] or OWL[3]. In addition, MDA-based languages do not have a knowledge based foundation to enable reasoning (e.g., for supporti quality assurance), which ontologies provide [2]. Beyond t ditional data models, like UML class diagrams or entity re tionship diagrams, ontologies provide methods for integra fragmented data models into a common model without l the notation and style of the individual models [14].

Seidenberg and Rector [22] proposed web ontolog mentation to counter decreasing ontology performanc ontology size increases. The algorithm to make ontolog mentation is similar to our approach, but we extend th of ontology areas for more stakeholders and volatilitie

---

[2] Resource Description Framework: http://www.w3.org/RDF/

[3] Web Ontology Language: http://www.w3.org/2007/OWL

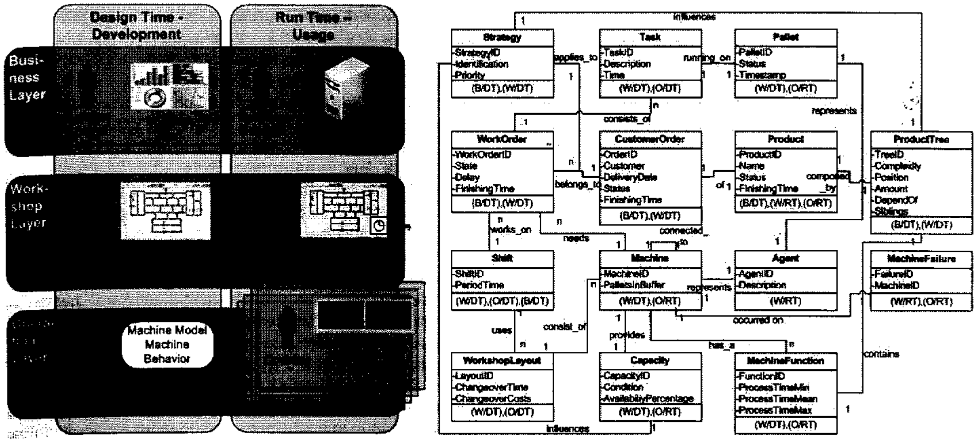

Figure 1: Sources of semantic gaps between stakeholders: domain layers, design-/run-time views; the data model contains common domain concepts to bridge semantic gaps.

III. Industry Use Case

In cooperation with industry partners in the production automation domain we conducted the project "Simulator for Assembly Workshops" (SAW) [15], which simulates complex reconfigurable production automation systems by scheduling sequences of transport and machine tasks over 100 times faster than the actual hardware[4]. The SAW simulator has been lidated with real hardware components to ensure simulation dity for real-world production automation systems. In the W context stakeholders from different backgrounds work her and could benefit from better automated access to others data models which is currently only possible via keholders themselves as the data models are not well ted.

re (1 illustrates sources of semantic gaps between lders: stakeholder domain layers with different local gies; and design-/run-time views which are semanti- well connected. The data model, in our case an on- odel (the Engineering Knowledge Base (EKB) [17]), common domain concepts to bridge the semantic en stakeholder terminologies and design-/run-time

stakeholder layers in Figure 1 are: a) the *business* production planning to fulfil customer orders by nal work orders to the workshop; b) the *work-* W) for coordinating the complex system of trans- and machines to assemble smaller basic prod- more comprehensive products according to the d c) the *operation layer (O)* for monitoring the ort system elements and machines to ensure ns to the workshop tasks. Those three layers are divided into two parts based on the time those layers worked on, namely design time (development) and run time (usage).

Figure 1 (right hand side) illustrates part of the data model that represents common domain concepts for the uses cases in UML-class-diagram style notation. The bottom box of each data element shows which stakeholder layer (B, W, and O) needs this data element to conduct their tasks and when: at Design Time (DT) or Run Time (RT).

From the SAW project we derived the following use cases that illustrate semantic gaps between stakeholders and how to overcome these gaps using ontology-based approaches.

UC-1. Translation between local stakeholder terminologies. The business manager on the business layer receives customer orders and schedules work tasks to the coordinator in the workshop layer. While they have a defined interface for exchanging work task information, they use local terminologies for concepts that are only occasionally needed to resolve scheduling issues, e.g., reference to specific customer orders if limited workshop capacity does not allow to fulfil all work tasks in a shift and negotiation on which tasks have higher priority are necessary to determine which customer orders will be fulfilled. Because the stakeholders use different terminologies, translations are necessary to automate references to customer orders between stakeholders in business and workshop layers.

UC-2. Run-time measurement data representation and analysis for design model improvements. If an engineering knowledge base is available to support run-time decisions with design knowledge, it is easy to also provide all kinds of run-time measurements linked to design elements, e.g., actual capacity of infrastructure, to iteratively improve the accuracy of design estimates with feedback from run time.

---

[4] ol Institute; http://www.acin.tuwien.ac.at

IV. RESEARCH ISSUES

The general idea of Ontology Areas (OAs) is to structure a comprehensive ontology into smaller building blocks with the following benefits for the designer and user of the ontology:

- A *smaller ontology* based on OAs that contains the minimal necessary knowledge for a specific task can be selected from a comprehensive ontology to facilitate more efficient use and change.
- We expect a smaller ontology (consisting of selected OAs) to exhibit *lower cognitive complexity* for designers who work with ontologies to make tools that support the automation of stakeholder tasks.
- Specific OAs can contain the more volatile ontology elements and thus make the design of the overall ontology *more stable against changes*.

As measurement criteria for evaluation we use the size of an ontology (and an OA) by counting the number of facts and relationships. In our study context the comprehensive ontology consists of: a) the production automation domain concepts (i.e., data model in Fig. 1) for design-time and run-time elements; and b) stakeholder extensions to the data model, such as local terminologies and mappings, for all stakeholders.

We used the following guidelines to design the OAs: a) concepts that a particular stakeholder (in business, workshop, or operation layer) needs to fulfil his typical tasks in order to achieve cohesiveness of the OAs; b) discern between common domain concepts and local add-ons of a stakeholder (such as terminology), which may change in different project contexts; c) keeping apart more stable design-time concepts from more volatile run-time concepts; and d) structuring volatile run-time data by manageable time intervals depending on the frequency of data elements' change. According to these guidelines examples for concrete OAs are: the design-time concepts of a business stakeholder and the run-time terminology of a workshop stakeholder.

From the use cases we derive the following research issues (RIs) to investigate the benefits of an ontology structured with OAs compared to an ontology without OAs.

UC-1. Translation between local stakeholder terminologies. The ontology supports each role by allowing to use their local terminology to communicate with other stakeholders. For this task sufficient OAs need to contain for the communicating stakeholders: the common domain concepts in their universe of discourse (see also in Fig. 1 the data elements and their link to associated stakeholders), local terminologies, mappings between local terminology elements and common domain concepts (on class level).

RI-1a: Compare the *complexity* (size) of the minimal ontology with OAs to the complexity of the overall ontology in the study context.

RI-1b: Compare the *efficiency* of the minimal ontology with OAs to the efficiency of the overall ontology in the study context to conduct the translation task.

The other use cases address benefits from making links between design-time and run-time data elements available at run time.

UC-2. Run-time measurement data representation and analysis for design model improvements. In the study context the collection of run-time data points, e.g., on process characteristics and quality of service of the infrastructure, helps to provide data for future design improvements, e.g., for more realistic planning and more efficient system configurations. The designers and quality management personnel, who conduct the data analysis procedures, often do not know in advance precisely which analysis functions they will need. Thus, a considerable amount of raw data would be beneficial to store in the ontology for querying design-time relationships and run-time data together. Unfortunately, even moderate data collection (10 data points) at reasonable frequency (e.g., one measurement every second) leads over a shift of 8 hours to a number of run-time data elements that easily exceeds the size of the design-time data elements in the ontology.

OAs that are designed to hold all measurement instances of a data element in a certain time interval (e.g, one minute) allow to keep the complexity of the ontology needed for analysis manageable: Only the OAs that contain relevant run-time measurements for a given analysis need to be considered.

RI-2a: Determine the *minimal complexity of OAs* to support a specific data analysis task more efficiently, such as calculating process characteristics. Compare the result with OAs to the (cognitive) complexity of using a whole ontology.

RI-2b: Compare the *efficiency* of the minimal ontology with OAs to the efficiency of the overall ontology in the study context to conduct the data analysis task

V. ONTOLOGY AREAS FOR BRIDGING SEMANTIC GAPS

In this Section we explain in more detail how to address the use cases with an ontology that uses OAs as basis for the evaluation of the RIs in Section 6.

An ontology area is a subset of ontology as a building block that can solve a certain task. The ontology can be broken into ontology areas based on several aspects, for example by the time, volatility, layer and roles. Figure 1 shows the break down of ontology into several ontology areas based on the stakeholder layers (business, workshop, operation) and time when models are mostly used (design time and run time). Some parts of the data mode are much more volatile than others, e.g., run-time process measurements compared to design-time workshop layout. For example, each data point measu[illegible] once a second in a shift that takes 8 hours produces arou[illegible] 30,000 data point instances, which need to be reduced by s[illegible]tistical methods or will take considerably storage space.

To make an OA from the whole ontology, we can fo[illegible] this basic algorithm. First, define a task that is needed to [illegible] solved by the stakeholder. Second, find related classes [illegible] doing the task. Third, find classes that linked to the clas[illegible] step two. Fourth, drop other classes that are not needed [illegible] save as a new ontology. Also, we can reconstruct the [illegible] ontology from the ontology areas, by merging them to[illegible] into one ontology by using ontology tool like Protégé.

We illustrate in three use cases (UC-1 to UC-2) how [illegible] help reduce the complexity of the ontology for bridg[illegible] mantic gaps in production automation systems.

UC-1. Translation between local stakeholder terminologies. The stakeholders of the production automation systems need to work together to achieve their goal. A common data schema is not possible because the stakeholders usually use different data formats, local terminologies and tools to access the data from the system. The ontology (EKB – Engineering Knowledge Base) plays a role as a common domain concept, where the local terminologies from the stakeholders will be mapped to. By mapping each local terminology to the ontology, the system can translate the local terminologies from one stakeholder to the other stakeholders. The translation could be the name of function, some names in the argument of the function, different data format, or the meaning of some parameters. However, the complexity of the ontology may increase when the number of the terminologies and the stakeholders is also increases, since the ontology should store all terminologies, the mappings and the common concepts.

By using the ontology areas, the stakeholder can take a small part of the ontology that he really cares and solving his task with the same results but less complexity than by using the full ontology. The example is illustrated on figure 2.

Figure 2. Translation between Business Terminology and Workshop Terminology.

The business stakeholder has a local terminology "ClientContract", while the workshop stakeholder has a local [illegible]minology "BusinessOrder". Both have a common con[illegible] to class CustomerOrder in the Ontology Areas. Then, [illegible] terminologies will be mapped to the class CustomerOrder [illegible]ntion in Listing 1a.

[illegible] 1a. Mapping terminologies to the common concept.

```
g('ClientContract','CustomerOrder').
g('BusinessOrder','CustomerOrder').
```

[illegible] the mappings above, we can have a translation be[illegible] local terminologies by using rule on Listing 1b. [illegible] and result can be seen on Listing 1c.

Listing 1b. Simple translation rules.

```
(Term1,Term2) :-
erm1,CommonConcept),
erm2,CommonConcept),
 Term2).
```

Listing 1c. Translation result.

```
2).
ontract'
rder'
```

The translation is just one example for translations in general. OAs for this use case would just consider the parts of the ontologies for the stakeholders involved (see Figure 2): stakeholder concepts, their local terminologies and mappings, which can more easily be added to and removed from an ontology as stakeholders change in a particular context. The evaluation for this use case will be explained on section 6.

UC-2. Run-time measurement data representation and analysis for design model improvements. Run-time measurement information can be used to make design time information more accurate. Volatile information like run-time measurement can produce large amounts of data which would make a single ontology unnecessary large and slow down the performance of the ontology. The need for storing a high volume of run-time measurement data in the ontology occurs if the concrete future statistical analysis procedures are not known at the time of measurement.

Partitioning of the ontology in areas of similar volatility allows building partial ontologies for the task or query at hand. Run-time measurement at the frequency of 1 data point per second provides 30,000 data points of shift of 8 hours. If this is too much information for the ontology to hold, it is possible to define OAs for smaller time windows, which allow including the data for a certain time frame to be loaded into the ontology for data analysis as needed without exceeding the capacity of the ontology.

Semantic gaps between run-time measurement and design-time information occur when we have data elements from the interface of the machine at run time, but there is no machine-understandable documentation for the design of the interface. To solve this problem, we first give meaning to run-time data that are needed to be stored in the ontology and then provide a link from run-time to design-time semantics.

For example, to find out the maximum process time of certain machine functions, we can measure the process duration of that machine function in one shift, so we collect sufficient and still manageable data. The measurement result is an event named "process" that consists of the id, the batch number, status and timestamp of machine function. Listing 2a shows several measurement results that can be obtained by filtering run time data. The real data themselves is a very long list.

Listing 2a. Run-time event data with semantic annotation.

```
% process (machine function id, batch number, status,
timestamp)
process('MF1','B-100','start',2009-02-03 T 10:01:06.01)
process('MF1','B-100','stop',2009-02-03 T 10:01:06.11)
process('MF2','A-200','start',2009-02-03 T 10:01:06.12)
process('MF1','B-101','start',2009-02-03 T 10:01:06.13)
process('MF1','B-101','stop',2009-02-03 T 10:01:06.21)
process('MF2','A-200','stop',2009-02-03 T 10:01:06.24)
```

To calculate the maximum process time of certain machine function, first we should calculate each process time by using predicate "process_time" to find the difference between the timestamp of "stop" status and the related timestamp of "start" status from the same machine function and batch number, and the keep it in the list using "list_of_process_time" predicate. Then with using the predicate "maxprocess" we will find the maximum value of process time of certain machine function (MFun) from the list of process time.

Listing 2b. Example analysis rule on run-time data.

```
max(X,Y,X) :- X >= Y.
max(X,Y,Y) :- X < Y.
maxlist([X],X).
maxlist([X,Y|Tail],Max) :-
maxlist([Y|Tail],MaxTail),max(X,MaxTail,Max).
process_time(MF,SN,T) :-
  process(MF,SN,start,X),
  process(MF,SN,stop,Y),
  T is Y - X.
list_of_process_time(List,MFun) :-
findall(T,(process_time(MF,SN,T),MF = MFun),List).
maxprocess(MFun,T) :-
  list_of_process_time(List,MFun),
  maxlist(List,T).
```

For query, for example we want to know the maximum process time of 'MF1'. The result can be seen on Listing 2c.

Listing 2c. Result of data analysis.

```
maxprocess('MF1',T).
T = 0.1
```

The machine function entity in design time consists of the id and process time attributes. Usually the values of process time attributes come from estimation, but by using run-time measurement on process time, we can compare the previous design-time estimates to actual run-time data analysis for research on design improvements.

The illustrating example above is simple enough to conduct statistical analysis at run time, but for more complex statistical analyses, a solution for storing large amounts of data in an ontology may be necessary, which would inflate ontology size and decrease the ontology reasoning performance. OAs allow to manage stacks of run-time data elements and keep the size of ontology within well-performing capacity ranges.

VI. Evaluation and Discussion

We have implemented the OAs from the SAW ontology using Protégé 3.3.1. The SAW ontology consists of 24 classes and 3,000 instances from the simulation of production automation system. The evaluation will compare the measurement of the whole ontology and the ontology areas for three different use cases explained in section 5, as follows.

UC-1: Translation between local stakeholder terminologies. We compare the complexity (size) of the minimal ontology with OAs to the complexity of the overall ontology in the study context. For the minimal ontology with OAs, the business and workshop stakeholders have local terminologies of 300 and 400 words, respectively. Both need 100 words to communicate with each other. There are 200 to 700 data elements representing common knowledge, and 200 words for mapping from both local terminologies to the common concepts. Totally 1,100 to 1,600 entities are needed for the OAs.

Meanwhile, the comprehensive ontology for 6 stakeholders consists of around 1,800 words for local terminologies and around 300 words to communicate with each other. There are 1,600 words of common knowledge, and 600 to 1,800 words for mapping of all local terminologies to common concepts. In total, the comprehensive ontology consists of 4,200 to 5,400 words. In this case, OAs can reduce the ontology size to 20 to 30 % of the comprehensive ontology.

We can compare the efficiency of the minimal ontology with OAs to the efficiency of the whole ontology in conducting the translation task as follows. To produce 100 words of translation results from 200 words of mapping, the OAs needs 3 operators of query applying to those mapping.

The comprehensive ontology can produce more translations (300 words) with 3 operators of query as well. But the query should be applied to more mapping (600 to 1,800 words). With OAs we can reduce the size of mapping and make the operation faster.

UC-2: Run-time measurement and analysis for design improvement. For evaluation we will determine the minimal complexity of OAs to support a specific data analysis task more efficiently, such as calculating process characteristics. Then we will compare the result with OAs to the (cognitive) complexity using a comprehensive complexity.

In the OAs of the specific task, for 1 volatile entity the run-time measurement consists of 30,000 data points per shift. In the overall ontology, there may be many more, e.g., 300,000, data points in one shift. By using the OAs, the user can focus only on entity that he needs, and thus reduce the complexity of data handling considerably.

The efficiency of the minimal ontology with OAs is compared to the efficiency of the overall ontology in the case to conduct the data analysis task as follows. In the OA, to obtain 5 data points analysis, it needed to run 3 operators of query over 30,000 data points at one shift. Hence 18,000 operations on data points are needed to obtain one of the measurements.

In the whole ontology, to obtain 20 data points analysis, it needed to run 3 operators of query over 300,000 data points at one shift. Hence 45,000 operations on data points are needed to obtain one of the measurements. OA is notably more efficient than overall ontology.

Lesson learned. From the experiences with these use case[illegible] we can learn the following lessons.

Building a smaller ontology for a task. As OAs allow [illegible]cusing on the content of interest for a stakeholder task, [illegible] could show that the resulting ontology is considerable smal[illegible] A smaller ontology is often also more efficient to handle [illegible] allows tackling tasks that use a particularly large numb[illegible] data elements (e.g., run-time measurements in UC-3).

Focus stakeholders on relevant data elements. The co[illegible]nation of OAs, design-time, and run-time data elemen[illegible] lowed filtering relevant data elements for stakeholders, [illegible] would not be possible without the combination. Thus [illegible] approach helped lower the cognitive complexity fo[illegible] holders by providing just the relevant subset of the [illegible]hensive ontology.

Version management for ontology areas. With the [illegible]cept we can flexibly build task-oriented ontologies [illegible] different criteria (like volatileness, layers, roles). [illegible] possible to compare different versions of the same [illegible] production automation system designed with different parameter settings) to compare the run-time reactions to from changing design parameters. However, this ability also raises the need for better version management for OAs to ensure the building of consistent ontologies for specific tasks.

VII. Conclusion And Further Work

Ontologies support the translation between stakeholder local terminologies via common domain concepts, in our case production automation concepts. Typically, the ontology models become very large and complex compared to the basic data model (such as used in a data base to automate run-time processes) if they include several aspects on a domain and some parts of the data model are volatile. In this paper, we proposed a data modeling approach based on ontology building blocks, so-called "Ontology Areas" (OAs), which allow solving tasks with smaller parts of the overall ontology. We evaluated the proposed approach with use cases from the production automation domain. Major result in the study context is that OAs improved the efficiency of data collection task for decision making by lowering the cognitive complexity for designers and users of the ontology.

Further work. We see further research in the following directions.

Effort for OA design and use. While OAs make a comprehensive ontology, which stores and uses engineering knowledge both at design time and run time, more manageable, their application needs the effort of designers for structuring the overall ontology and for building task-specific smaller ontologies. Thus we will conduct empirical studies on the effort needed to design and use ontologies with OAs. Future work could include human-subject experiments to [illegible] complexity and efficiency more rigorously.

[illegible]*lines for the OA approach.* While we found OAs use-[illegible] manage a large and complex ontology, we see the need [illegible]delines for the application the OA approach when de-[illegible] a new ontology as well as for structuring already es-[illegible]ontologies with OAs to improve their performance.

[illegible]*nce effort.* Particularly for ontologies which [illegible]nged by many users concurrently, we see a po-[illegible]age of the concept of OAs, as areas with differ-[illegible] change can be easily separated, simplifying the [illegible]models for consistency etc. In the context of our [illegible]is could be measuring the effort for typical [illegible] as a new workshop layout, new machines, or [illegible] between machines.

[illegible] Acknowledgment

[illegible] our colleagues at ACIN and TU Prague, [illegible] and inspiring discussions; and the SAW [illegible] for providing the application environment [illegible] case.

References

[1] American National Standard, "Enterprise-Control System Integration," in *Part 1: Models and Terminology*. vol. ANSI/ISA-95.00.01-2000 North Carolina, USA: ISA (the Instrumentation, Systems, and Automation Society), 2000, p. 142.

[2] K. Baclawski, M. K. Kokar, P. A. Kogut, L. Hart, J. Smith, J. Letkowski, and P. Emery, "Extending the Unified Modeling Language for Ontology Development," *International Journal of Software and Systems Modeling (SoSyM)*, vol. 1, pp. 142-156, 2002.

[3] K. Balasubramanian, A. Gokhale, G. Karsai, J. Sztipanovits, and S. Neema, "Developing Applications Using Model-Driven Design Environments," *COMPUTER*, pp. 33-40, 2006.

[4] S. Bergamaschi, S. Castano, and M. Vincini, "Semantic integration of semistructured and structured data sources," *SIGMOD Rec.*, vol. 28, pp. 54-59, 1999.

[5] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins, "What are ontologies, and why do we need them?," *Intelligent Systems and Their Applications, IEEE [see also IEEE Intelligent Systems]*, vol. 14, pp. 20-26, 1999.

[6] I. R. Cruz, X. Huiyong, and H. Feihong, "An ontology-based framework for XML semantic integration," in *International Database Engineering and Applications Symposium (IDEAS '04)*, 2004, pp. 217-226.

[7] A. Doan, N. F. Noy, and A. Y. Halevy, "Introduction to the special issue on semantic integration," *SIGMOD Rec.*, vol. 33, pp. 11-13, 2004.

[8] D. Fensel, *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*: Springer, 2003.

[9] E. H. Gail, L. David, C. Jeromy, re, N. Fred, C. John, and N. Martin, "Application servers: one size fits all ... not?," in *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, Anaheim, CA, USA, 2003.

[10] A. Gangemi, N. Guarino, C. Masolo, and A. Oltramari, "Sweetening WordNet with DOLCE," *AI Magazine*, vol. 24, pp. 13-24, 2003.

[11] C. H. Goh, "Representing and Reasoning about Semantic Conflicts in Heterogeneous Information Systems." vol. PhD: MIT, 1996.

[12] T. R. Gruber, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing," in *Formal Ontology in Conceptual Analysis and Knowledge Representation*, N. Guarino and R. Poli, Eds.: Kluwer Academic Publishers, 1993.

[13] A. Halevy, "Why your data won't mix," *Queue*, vol. 3, pp. 50-58, 2005.

[14] M. Hepp, P. De Leenheer, A. De Moor, and Y. Sure, *Ontology Management: Semantic Web, Semantic Web Services, and Business Applications*: Springer-Verlag, 2007.

[15] M. Merdan, T. Moser, D. Wahyudin, and S. Biffl, "Performance evaluation of workflow scheduling strategies considering transportation times and conveyor failures," in *International Conference on Industrial Engineering and Engineering Management (IEEM 2008)*, 2008, pp. 389-394.

[16] J. Miller and J. Mukerji, "Model Driven Architecture (MDA)," *Object Management Group, Draft Specification ormsc/2001-07-01, July*, vol. 9, 2001.

[17] T. Moser, A. Schatten, W. D. Sunindyo, and S. Biffl, "A Run-Time Engineering Knowledge Base for Reconfigurable Systems," Institute for Software Technology and Interactive Systems, Vienna University of Technology, Austria, Vienna, 2009, http://tinyurl.com/c5snc3 .

[18] I. Niles and A. Pease, "Towards a standard upper ontology," in *2nd International Conference on Formal Ontology in Information Systems*, 2001, pp. 2-9.

[19] N. F. Noy, "Semantic integration: a survey of ontology-based approaches," *SIGMOD Rec.*, vol. 33, pp. 65-70, 2004.

[20] N. F. Noy, A. H. Doan, and A. Y. Halevy, "Semantic Integration," *AI Magazine*, vol. 26, pp. 7-10, 2005.

[21] A. W. Scheer, *Computer-Integrated Manufacturing*, 4th ed.: Springer, 1989.

[22] J. Seidenberg and A. Rector, "Web Ontology Segmentation: Analysis, Classification and Use," in *International World Wide Web Conference (WWW 2006)* Edinburgh, Scotland: ACM, 2006, p. 10.

[23] M. Uschold and M. Gruninger, "Ontologies and semantics for seamless connectivity," *SIGMOD Rec.*, vol. 33, pp. 58-64, 2004.