R. Reussner, A. Koziolek, R. Heinrich (Hrsg.): INFORMATIK 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2021 939

Modelling of Interaction in Interweaving Systems as Ontology Mapping Adaption

Matthias Jurisch[1], Bodo Igler[1]

Abstract: Interweaving Systems (IWS) are systems that have not been designed to cooperate, but can influence each other at runtine through a shared environment. In this paper, we present a general approach that enables IWS to gain an explicit view of the shared environment. The approach is based on ontology alignment and model driven techniques. The core idea is to build a *Directory Ontology* that shows what systems can know about their environment, and local *Connector Ontologies* that organize information routing between systems and connect these via ontology alignments. Changes are addressed using ontology mapping adaption. This approach is presented using an application example from the area of traffic control systems. The key benefit of the approach is that it allows supporting and sustaining data integration in Interweaving Systems using explicit and domain-independent rules.

Keywords: Interweaving Systems; Ontology-Driven Development; Ontology Mapping Adaption

1 Introduction

Interweaving Systems [To16] are systems that have not been designed to cooperate, but can influence each other at runtime by interacting through a shared environment. Interweaving systems work under conditions that require some kind of soft or hard real-time constraints. Therefore, they are first and foremost optimized to satisfy these constraints. Optimization for domain-specific goals is usually regarded as less important. However, deliberate cooperation between interweaving systems can potentially improve these soft aspects without touching the real-time/safety critical part.

Whether and how a shared view of the environments of Interweaving Systems improves cooperation and performance regarding domain-specific goals is still an open question. A first step to address this question consists of designing and evaluating approaches for environment sharing. An approach that allows sharing of data between interwoven systems would change the nature of interactions from accidentally interwoven systems to deliberate interactions supported by a central authoritative knowledge, which might also support the systems in dealing with emergent stability issues. As the typical environments of Interweaving Systems are prone to changes and as parts of such systems can fail, appropriate approaches must consider these issues, too.

[1] Department of Design – Computer Science – Media, RheinMain University of Applied Sciences, Unter den Eichen 5, 65195 Wiesbaden, {matthias.jurisch,bodo.igler}@hs-rm.de

doi:10.18420/inf2020_86

In this work, we present a general approach to support Interweaving Systems in creating a shared view on their environment. It employs knowledge based techniques such as ontologies and inference. To allow systems to access the shared view, a modification of these systems is required. What kind of information a system can gather about the environment (e.g., what sensors it can use to observe it) is modeled via a *Directory Ontology*. Each system can view the Directory Ontology and use it to request more information regarding specific aspects of the environment. In this way, systems will only receive data concerning the environment that is relevant to them. What kind of information is relevant to a system is modeled in a local *Connector Ontology* that is connected to the Directory Ontology using ontology alignments. Changes in the environment and system failures require that the alignment is constantly adapted. This issue is addressed with a technique called Ontology Mapping Adaption [Gr13a]. The key benefit of the approach is that it allows supporting and sustaining data integration in Interweaving Systems using explicit and mostly domain-independent rules.

In our previous work, we presented a domain-specific technique that addresses this problem for interweaving systems in the domain of autonomous traffic-control [JI17]. The main contribution of this paper is twofold: (1) We present an abstract, i.e., domain-independent approach that shows how the domain-specific technique can be applied to interweaving systems in general. (2) We show how the case study of [JI17] fits into the application of the abstract approach. This includes an evaluation of how the approach needs to be tailored to the specific use case and which model transformations need to be implemented. We also demonstrate, how autonomous systems can use this data in a domain specific use case and how the shared view on the environment can be useful.

The remainder of this work is structured as follows: Section 2 gives an overview of the background and related work and identifies the research gap. An overview of the approach and the models that are involved and how these models are used to create a shared view is shown in Section 3. Section 4 describes an application and shows how the approach can be tailored to domain-specific problems. A discussion is given in Section 5. The paper ends with a conclusion and an outlook to future work in Section 6.

2 Background and Related Work

Interweaving Systems are systems that are designed in some kind of technical context that is characterized by several aspects: The systems in this context influence each other, both via *defined interactions* and *not defined interactions*. Also, the systems are heterogeneous in a sense that no central instance controls all of them and they were not designed to take system's influences into account. The environment of the systems in general is uncertain, but can be partly observed. This has severe consequences for their effectiveness, as determining the outcome of interactions becomes very difficult. Another important aspect is that the application domain requires the systems to operate under real-time conditions. [To16]

Interweaving Systems are often supported using so called organic computing approaches [MSSU11] and often focus on making systems themselves more fault-tolerant and more able to deal with changing environments. Organic computing techniques draw inspiration from biological systems and are commonly used in the area of self-managing systems. In this work, we approach the area of interweaving systems from a different perspective: We focus on allowing Interweaving Systems to communicate about their environment and making this environment explicit. The messages exchanged in this model contain information on what data is available. This communication is supported by formal conceptual models (ontologies). This leads to several areas of related work, including (1) Model Driven Software Engineering for communication middleware, since our approach can be regarded as belonging to this area (2) ontology-driven software engineering and (3) ontology alignment adaption.

In the area of Model Driven Software Engineering for communication middleware, several works have shown that model driven software development can be used to facilitate the usage of middleware by using models to represent properties of data and middleware aspects [Ed04, BT10]. While these approaches show that a model driven approach can significantly reduce the lines of code needed to be written, the models and the modeling languages are domain-specific and do not allow automated reasoning, which makes them less flexible. This shortcoming leads to the area of ontology-driven software engineering (2) that uses ontologies as a modeling language for model driven software engineering. One of the benefits of using ontologies in this context is that ontologies can be easily linked to other ontologies and the declaration of these links allows for an automatic generation of model transformations [Pa12]. While approaches from this area can be used to model systems working in a real-time domain [St17], these models are static and do not account for changes in the environment or aligned ontologies.

Reacting to changes in aligned ontologies is an area of research that has been approached using several methods: Both applying rules to changes directly [Gr13b, do15] as well as applying rules to changing inferences [Ju16] have been proposed. In addition to these rule-based approaches, machine learning approaches have been explored [JI19]. While a rule-based approach has been proposed for a use case in the area of Interweaving Systems, this approach is tied to the domain-specific aspects of the use case [JI17]. To our knowledge, no domain-independent approach exists which facilitates environment sharing based on ontologies. This issue is at the core of the research presented in this paper.

3 Approach

The main goal of the approach presented in this work is to allow interweaving systems to cooperatively create a shared view of their environment. This approach needs to fulfill a set of non-functional requirements: (1) it needs to be fault-tolerant in the sense that it can still support a shared view on the environment when some systems fail, and (2) it needs to be fault-tolerant in the sense that it allows dealing with partial communication failures, and (3) it must not interfere with real time properties of the interweaving systems. Such an approach

is beneficial for IWS which observe their environment and whose performance (including functional aspects) can be improved by sharing these observations. As environment sharing requires additional computational resources, the systems also need some kind of isolation between real-time critical aspects and the program that implements our approaches, for example implemented by a mixed-criticality system [BD13].

The core aspects of our approach consist of three main ideas: (1) An architecture for supporting data sharing in an interweaving systems context, (2) an implementation strategy for this model based on ontologies and (3) an algorithm that is used to adapt the ontology model to changes. This paper focuses on the first two ideas. Although the approach has to be adapted to the respective domain specifics, the core ideas are domain-independent. These ideas are discussed in the following subsections.

3.1 Architecture

A simplified example of an application of the framework with two systems is shown in Figure 1. For the sake of simplicity, ontologies are represented as storages, while they can also be implemented as a view on concepts that exist in multiple other storages. Pre-existing components of interweaving systems are shown in yellow, our additions to these concepts is depicted in blue with dashed lines. In this paper, we will focus on the Connector Ontology, the Directory Ontology and the reaction to changes. The models used in the framework for knowledge sharing mainly need to fulfill two purposes: (1) let systems discover what information on the environment is available (2) bind information from remote locations to local needs. The first purpose is implemented using a model called directory ontology. The Directory Ontology contains information on two aspects: It represents which interweaving systems are able to exchange data. Each system can provide a set of sensors that can be used to observe the environment. These sensors are also represented in the Directory Ontology.

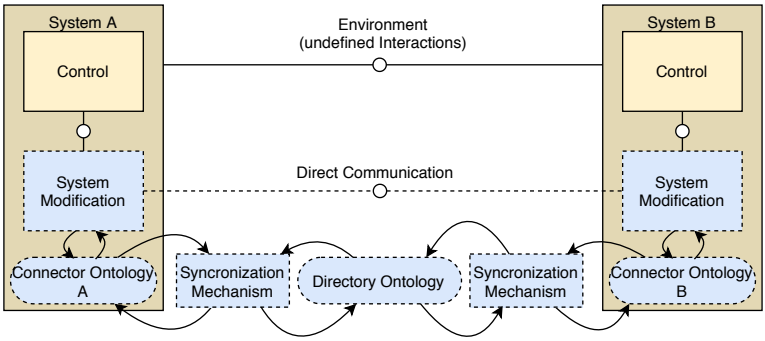

Fig. 1: Approach to Augmentation of IWS

The second purpose is fulfilled by a Connector Ontology for each system. The connector ontology, which is specific to each system, can label a sensor as local, or as a remote. Remote

sensors are deployed on other systems. For remote sensors, connection information is stored. An example for connection information can be a DDS topic ID or a URL. In the Directory Ontology, sensors can be related to data needs that are used to represent the requirements of data to improve the self-management capabilities of a system. The Connector Ontology also contains a Similarity Ontology that stores similarity scores. These scores are used to associate remote sensors to other stand-in sensors which can replace them in case of failing remote sensors. (This includes failure of other systems and communication failures, which lead to missing remote sensor data.) Pairs of sensors are assigned a similarity score that represents how well readings from a sensor are suited to replace readings from the other sensor.

A synchronization mechanism is used to integrate the Directory Ontology with local Connector Ontologies. This synchronization mechanism needs to fulfill two purposes: (1) ensure that the directory contains the most up-to-date information on sensors available on a system and (2) allow each system to access what sensors are available on other systems. Hence, this mechanism can be implemented by systems broadcasting their sensor capabilities in a predetermined interval. Based on this information, local systems can interact with each other to exchange data according to the definitions used in local and global ontologies. This information exchange can be implemented using a peer-to-peer framework such as DDS [Pa03] or any topic-oriented middleware that does not rely on a central broker. The same middleware can also be used to exchange Directory Ontology information.

3.2 Ontology Design

All models and, as far as possible, the change mechanism should be implemented using standardised ontology languages, so that standard tools can be used to process the models. Wherever possible, only subclass reasoning should be used. Subclass-based reasoning requires only low computational complexity. While the usage of subclass reasoning limits the expressiveness of the ontologies, the expressiveness suffices for the model presented in this paper.

These design guidelines lead to implementation strategies for the Directory and Connector Ontology. An example for these ontology types is shown in Figure 2. The class *System* models all sensor readings. Therefore, if a sensor is part of a system, the class representing the sensor is a subclass of the class representing the respective system. A sensor reading would be an instance in this ontology. The connector part of the ontology represents the *Need* concepts and shows what sensors are *External*. Every sensor that is a subclass of External is a sensor at a remote system that provides data to the local system. All connections between the connector and Directory Ontology are alignment connections. The Similarity Ontology part of the Connector Ontology, which is shown at the bottom of Figure 2, is based on *Similarity Entries* that relate pairs of local and remote sensors to similarity scores. Due to design choices regarding the Connector Ontology, Connector Ontology classes need to be represented as instances in the Similarity Ontology.

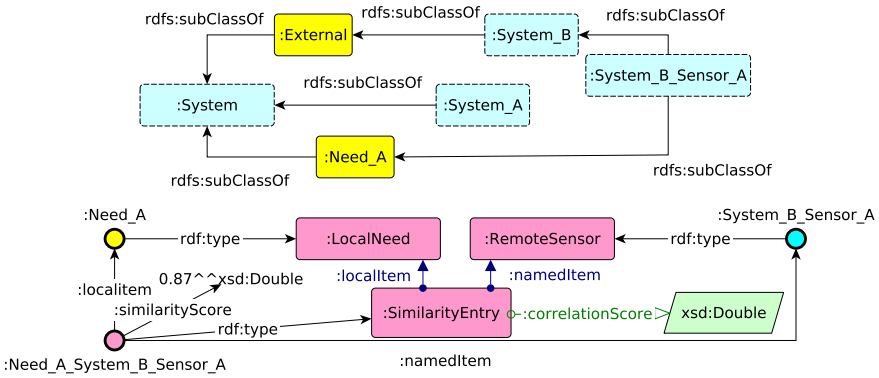

Fig. 2: Example Connector Ontology (yellow) with mapped Directory Ontology classes (blue with dashed lines) including Similarity Ontology (bottom)

3.3 Adapting to Changes

To adapt the mapping between Directory and Connector Ontologies when the Directory Ontology changes, we use a rule-based technique published in [Ju16, JI17]. The basic idea of this approach is to compute inferences before and after a change occurs. A comparison of the inferences leads to a set of deleted and added inferences named Δ_{inf}. Then two kinds of rules are applied to entries in Δ_{inf}: (1) rules that identify whether a change in the ontologies requires an adaption of the alignment and (2) rules that compute the actual changes to the alignment.

The first kind of rules can be implemented by simply searching for changes in inferences that are alignment statements. For each changed alignment statement, replacements need to be computed. This is the task of the second kind of rules: if an alignment statement needs to be adapted, the second kind of rule searches the Similarity Ontology for replacements using SPARQL queries. These replacements are then applied to the ontology alignment.

4 Transformation to a Domain-Specific Example

This section outlines how the general concept can be transferred to a concrete application in traffic control systems. A more detailed description of this application was published in [JI17]. In the example, traffic control systems controlling the traffic lights at one intersection have a fixed set of signaling plans that they can change based on traffic flow data. The approach is used to exchange traffic flow data between intersections, so that traffic control systems can incorporate traffic flow data measured at remote intersections into their decision making process.

Applying the approach requires some domain-specific artifacts to be created: change rules, the modification of the IWS and the incorporation of environment data into the traffic control process are always domain specific and need to be created for each use case. Other aspects are transformed in the sense of model driven software engineering. The specialization transformation is used to add the required domain-specific information to the abstract ontology model. In this case, the class *System* is used to represent traffic control systems. The class *Data* needs to be specialized so that it can hold data for traffic flow. Hence, a datatype is added to the class. *Data Needs* are used to represent data needs specific to intersection entries and exits. The remainder of the model remains unchanged. As described, when reacting to changes, rules are used to search Δ_{inf} for entries with instances of the class *External*. In this use case, a new alignment statement is generated so that the best candidate for a data need can be used. This process is based on a SPARQL query, that explicitly represents the change rule.

5 Discussion

The core contribution of this work is to present an approach that shows how to build a data structure that can use mapping adaption to deal with changes in the environment of interweaving systems. The approach supports the separation of concerns on several levels: domain-specific concepts are separated from domain-independent ones, and creating a shared model of the environment is separated from the regular functionality of the systems. Also, rules for changes, domain-specific rules and rules for connecting systems are separated. In addition, the approach allows for rules concerning the systems to be formulated in an explicit, short and domain-independent way. This also means that these rules are not hidden in the source-code of the system or as a part of assumptions of a domain-specific language.

6 Conclusion and Outlook

This paper shows, how an ontology-alignment-based approach can support data integration in interweaving systems and sustain integration when changes in the environment occur. Using these technologies allows a modularization of several aspects that are important to the scenario and an explicit formulation using standardized, domain-independent modeling languages. Future work could explore more application areas for the approach such as the industrial manufacturing domain. Also, the quantitative benefit of the approach could be evaluated and compared to other methods of systems management.

Bibliography

[BD13] Burns, Alan; Davis, Robert: Mixed criticality systems-a review. Department of Computer Science, University of York, Tech. Rep, pp. 1–69, 2013.

[BT10] Beckmann, Kai; Thoss, Marcus: A model-driven software development approach using OMG DDS for wireless sensor networks. In: IFIP International Workshop on Software Technolgies for Embedded and Ubiquitous Systems. Springer, pp. 95–106, 2010.

[do15] dos Reis, Julio Cesar; Pruski, Cédric; Silveira, Marcos Da; Reynaud-Delaître, Chantal: DyKOSMap: A framework for mapping adaptation between biomedical knowledge organization systems. Journal of Biomedical Informatics, 55:153 – 173, 2015.

[Ed04] Edwards, George; Deng, Gan; Schmidt, Douglas C.; Gokhale, Aniruddha; Natarajan, Bala: Model-Driven Configuration and Deployment of Component Middleware Publish/-Subscribe Services. In (Karsai, Gabor; Visser, Eelco, eds): Generative Programming and Component Engineering. Springer, Berlin, Heidelberg, pp. 337–360, 2004.

[Gr13a] Groß, Anika; Dos Reis, Julio Cesar; Hartung, Michael; Pruski, Cédric; Rahm, Erhard: Semi-automatic Adaptation of Mappings between Life Science Ontologies. In (Baker, Christopher J. O.; Butler, Greg; Jurisica, Igor, eds): Data Integration in the Life Sciences. Springer, Berlin, Heidelberg, pp. 90–104, 2013.

[Gr13b] Groß, Anika; Dos Reis, Julio Cesar; Hartung, Michael; Pruski, Cédric; Rahm, Erhard: Semi-automatic Adaptation of Mappings between Life Science Ontologies. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 7970 LNBI:90–104, 2013.

[JI17] Jurisch, Matthias; Igler, Bodo: Knowledge-Based Self-Organization of Traffic Control Systems. In: 47. Jahrestagung der Gesellschaft für Informatik, Informatik 2017, Chemnitz, Germany, September 25-29, 2017. pp. 947–954, 2017.

[JI19] Jurisch, Matthias; Igler, Bodo: Graph-Convolution-Based Classification for Ontology Alignment Change Prediction. In (Mehwish Alam, Davide Buscaldi et al., ed.): Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG2019) Co-located with ESWC 2019, Portoroz, Slovenia, June 2, 2019. CEUR-WS.org, pp. 11–20, 2019.

[Ju16] Jurisch, Matthias: Managing Ontology Mapping Change based on Changing Inference Sets. In: Knowledge Engineering and Knowledge Management - EKAW 2016 Satellite Events. Springer, pp. 255–262, November 2016.

[MSSU11] Müller-Schloer, Christian; Schmeck, Hartmut; Ungerer, Theo: Organic Computing - A Paradigm Shift for Complex Systems. Springer-Verlag, Berlin, Heidelberg, 2011.

[Pa03] Pardo-Castellote, G.: OMG Data-Distribution Service: architectural overview. In: 23rd International Conference on Distributed Computing Systems Workshops, 2003. Proceedings. pp. 200–206, May 2003.

[Pa12] Pan, Jeff Z.; Staab, Steffen; Amann, Uwe; Ebert, Jürgen; Zhao, Yuting: Ontology-Driven Software Development. Springer Publishing Company, Incorporated, 2012.

[St17] Steinmetz, Charles; Schroeder, Greyce; dos Santos Roque, Alexandre; Pereira, Carlos Eduardo; Wagner, Carolin; Saalmann, Philipp; Hellingrath, Bernd: Ontology-driven IoT code generation for FIWARE. In: 2017 IEEE 15th International Conference on Industrial Informatics (INDIN). IEEE, pp. 38–43, 2017.

[To16] Tomforde, S.; Rudolph, S.; Bellman, K.; Würtz, R.: An Organic Computing Perspective on Self-Improving System Interweaving at Runtime. In: 2016 IEEE International Conference on Autonomic Computing (ICAC). pp. 276–284, July 2016.