COST Action IC1404: Multi-Paradigm Modelling for Cyber-Physical Systems

http://www.mpm4cps.eu/

MPM4CPS

Proceedings of the 4th Workshop of the MPM4CPS COST Action

September 15-16, 2016 • Gdańsk, Poland

Hans Vangheluwe, Vasco Amaral, Holger Giese, Jan Broenink, Bernhard Schätz, Alexander Norta, Paulo Carreira, Miguel Goulão, Antonio Vallecillo, Tanja Mayerhofer (Eds.)

Technical Report No. ITI16/01
Departamentos Lenguajes y Ciencias de la Computación Universidad de Málaga

Editors:

Hans Vangheluwe
University of Antwerp (Belgium)

Vasco Amaral
Universidade Nova de Lisboa (Portugal)

Holger Giese
Hasso-Plattner-Institut für Softwaresystemtechnik GmbH (Germany)

Jan Broenink
University of Twente (Netherlands)

Bernhard Schätz
fortiss GmbH (Germany)

Alexander Norta
Tallinn University of Technology (Estonia)

Paulo Carreira
Universidade de Lisboa (Portugal)

Miguel Goulão
Universidade Nova de Lisboa (Portugal)

Antonio Vallecillo
Universidad de Málaga (Spain)

Tanja Mayerhofer
TU Wien (Austria)

Table of Contents

Papers

Presentations

Preface

In virtually any area of human activity, Cyber-Physical Systems (CPS) are emerging. CPS are truly complex, designed systems that integrate physical, software and network aspects. To date, no unifying theory and no systematic design methods, techniques and tools exist for such systems. Individual mechanical, electrical, network or software engineering disciplines only offer partial solutions. Multi-paradigm Modelling (MPM) proposes to model every part and aspect of a system explicitly, at the most appropriate level(s) of abstraction, using the most appropriate modelling formalism(s). Modelling language engineering, including model transformations, and the study of their semantics, are used to realize MPM. MPM is seen as an effective answer to the challenges of designing CPS.

The COST Action IC1404: Multi-Paradigm Modelling for Cyber-Physical Systems (MPM4CPS) aims to promote foundations, techniques and tools for multi-paradigm modelling for cyber-physical systems, and to provide educational resources to both academia and industry. This will be achieved by bringing together and disseminating knowledge and experiments on CPS problems and MPM solutions.

This workshop was the fourth workshop held in the context of this COST Action. It was held co-located with the Federated Conference on Computer Science and Information Systems 2016 (FedCSIS) on September 15-16 in Gdańsk, Poland. The program comprised presentations of MPM4CPS COST Action members discussing their work on foundations, techniques and applications of MPM4-CPS, as well as joint work meetings. These proceedings collect the presentations given at the workshop, as well as selected papers detailing the presented work. Please note that the collected papers represent non peer-reviewed work in progress carried out by the COST Action members in the area of MPM4CPS.

The presentations and papers collected in these proceedings cover many different aspects of multi-paradigm modelling for cyber-physical systems including, but not limited to

- tools and techniques in MPM4CPS including
  - system architecture modelling,
  - system integration,
  - system composition and decomposition,
  - system verification,
- classifications of formalisms and
- industrial practices and ecosystems.

We would like to thank the presenters and paper authors contributing their work to this COST Action. Furthermore, we would like to thank Maria Ganzha and Marcin Paprzycki for organizing the workshop in Gdańsk, as well as the Polish Information Processing Society and the IEEE Poland Section Chapter of the Computer Society for their support.

November 2016

Model-Driven Technical Space Integration Based on a Mapping Approach

Vladimir Dimitrieski, Slavica Kordić, Milan Čeliković, and Ivan Luković

University of Novi Sad, Faculty of Technical Sciences,
Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia
{dimitrieski,slavica,milancel,ivan}@uns.ac.rs

Abstract. In this report we propose a research with a goal to create a "smart" integration approach and tools that will alleviate integration problems that currently exist in the domain of Industry 4.0. First, the main building blocks of Industry 4.0 and Cyber-Physical Systems are described. Next, the main motivation and goals behind the proposed research are introduced along with a presentation of possible real-world applications. In addition to the research proposal, we also present a detailed literature analysis on the topics of schema matching, mapping and ontology alignment that are closely related to the domain of data and device integration.

Keywords: Integration, Mapping, Model-Driven, Technical Space, Domain Specific Language, Cyber-Physical Systems

1 Introduction

In this report we propose a research that aims to provide a solution or at least alleviate integration problems that currently exist in the domain of Industry 4.0. In order to understand these problems and their repercussions, we need to present current manufacturing trends, such as Industry 4.0, and put them into a historical context. Afterward, we present main Industry 4.0 components with the emphasis on manufacture automation and how it relies on the integration of these components.

Manufacturing has been the driving factor behind the development of human race since its inception. The manufacture of things for a specific use began with the production of basic necessities and household items well before 4000 B.C. [79]. The end products were simple as well as the manufacturing process that usually utilized basic materials such as wood, stone, or metal. Over the following centuries, the manufacturing process gradually improved and these simple production steps steadily began to develop into better and more complex operations.

Although the manufacturing process developed at a more or less steady pace over the course of history, several sudden and significant paradigm changes happened when the whole process was greatly influenced by new inventions. These sudden shifts or improvements of the process are known as “industrial revolutions” (Figure 1). The trigger for the first industrial revolution was the invention of the steam engine by James Watt in 1784. A domination of manual labor was disrupted by the increasing mechanization which generated greater output of the produced goods and increased their quality by mitigating human errors and shortening time needed for products to reach its consumers. The subsequent revolutions were also caused by inventions that allowed even greater degree of automation in the manufacturing process. In the 1870’s, the electrical energy and the introduction of the assembly line paved the way for mass production of goods. In 1969, the first programmable logic controller (PLC) was created and the digitalization began to infiltrate the manufacturing process as well as all other aspects of life. Such a widespread digitalization provided means for better and smarter machines with the aim to slowly decrease human participation in the manufacturing process. However, machines still had to be operated by humans and, as such, were not fully independent and self-adjustable to variations in the manufacturing processes.

In the resent years, a new paradigm shift is happening and it is enabled by the advances in digitalization. The shift is dubbed the 4th industrial revolution or Industry 4.0 in short. Industry 4.0 promises to improve operational effectiveness, develop entirely new “smart” services and products, as well as new business models [77]. The term Industry 4.0 (ger. *Industrie 4.0*), was coined in 2011 when Kagermann et al. promoted ideas on how to strengthen the competitiveness of German manufacturing industry [78]. The term has been later used in “High-Tech Strategy 2020” initiative of the German Government and has become eponym for all high-tech projects to be implemented by the 2020.

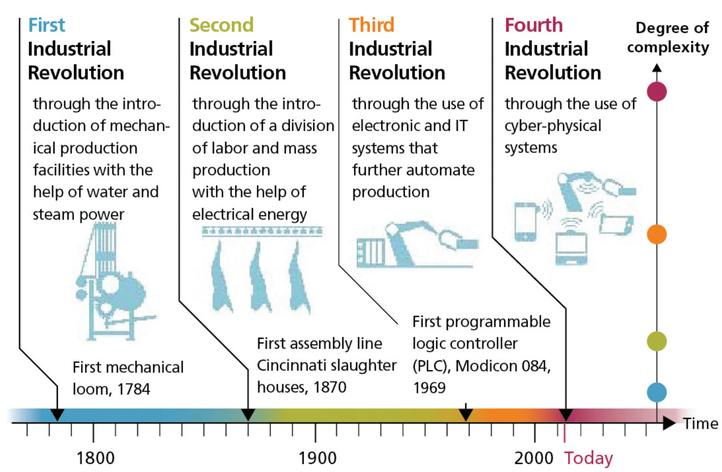


Fig. 1. Industrial revolutions (source [106])

The term Industry 4.0 has been coined in Germany and its main stakeholders come from this country. Outside of Germany, similar ideas and vision may be found under the names Industrial Internet and Advanced Manufacturing [45,68]. The Industrial Internet, also called Industrial Internet of Things (IIoT), has been introduced by General Electric (GE) [60], and later put under supervision of Industrial Internet Consortium (IIC) where GE was joined by many private companies and academic institutions around the world making the IIoT a global movement. Both Industry 4.0 and IIoT encompass the same vision where machinery, people, and analytic are tightly tied together. However, unlike Industry 4.0 which focuses on manufacturing processes, the IIoT stretches beyond manufacturing and enters the sectors such as energy, transportation, healthcare, and agriculture [26]. A part of the IIoT that only focuses on the manufacturing sector was named Advanced Manufacturing in [89].

1.1 A Brief Overview of Industry 4.0

The idea of smart products and smart machines in the context of manufacturing is not new. Computer Integrated Manufacturing (CIM) was a vision of 1980's, where the complex, state-of-the-art computers were introduced in factories with a goal to fully automate the production and solve cost and product quality problems that were very pronounced in the manufacturing process [64, 156]. The vision of human-less factories soon was shattered by reality in which CIM systems were extremely complex in planning as well as in construction, operation, and maintenance [164]. The technologies were not yet mature and the humans were overworked. However, the vision of fully automated and computer-centered manufacturing continued to live and evolve through the evolution of technology. The focus started to move away from big, clunky super computers that drive the production, to smart, independent computers embedded into every aspect of the manufacturing process. The basics of the omnipresent technology idea were introduced by Weiser [159] in 1991. He envisioned the world of ubiquitous computers in which computers are *"weaved into the fabric of everyday life until they are indistinguishable from it"*. Weiser, and later Poslad in [125], stated that one of the main requirements for adoption of ubiquitous computers is that they are context-aware. Context-aware computers are able to provide up-to-date and relevant information about the environment and state they are in. As it often happens with the technology, what was tested in everyday life once it reaches a mature state is introduced in the manufacturing process if it can improve it. Therefore, once the appropriate level of computer maturity was reached, ubiquitous computing slowly emerged as a main element of modern manufacturing process and currently is one of the main enablers of Industry 4.0.

Industry 4.0 is driven by the Internet, increasing number of connected devices, and future-oriented technologies for the implementation of smart machines and products. More than ever before, fast development cycles, flexibility, resource efficiency, decentralization of production, and individualization on demand are in the spotlight of the manufacturing [50, 90]. Market has changed and companies not only have to be the first to the market with their own product but they also have to provide a high degree of customization thus adapting their products to the needs of individual buyers. This often requires the production in “batch of one”, where the mass produced goods are individualized and customized in order to better suit buyers’ needs. This leads to the market shift from sellers to buyers, where buyers are conducting trade on they own terms. Individualization of products has become one of the main selling points for most of the companies. Additionally, as products need to be introduced to a market as soon as possible, innovation, and development periods need to be shortened.

The vision behind Industry 4.0 is the creation of smart, modular, and efficient manufacturing systems in which the products control their own production. Products relay instructions and information about their current status and external conditions (i. e., production context) to the manufacturing system. Based on the received information, the system is able to adapt its behavior and perform necessary steps. For example, products could send a message to the system providing information in which production phase they are, what are the constraints of this particular object that a manufacturing system must comply with, what is the next step in the production process that needs to be performed, etc. This is supposed to allow mass production of products of the same type while complying with the constraints and customization requirements of each product.

In order to provide a definition of the term Industry 4.0, we must introduce main concepts that are building blocks of the Industry 4.0 vision. According to [68, 90, 155], the following fundamental concepts and terms may be identified (c.a. Figure 2):

– **Cyber-Physical System (CPS)** may be defined as a system where cyber and physical components are connected closely at all levels [18]. The term *cyber component* denotes a component used for discrete processing and communication of information, while the term *physical component* is used to represent a natural or man-made technical component that operates in

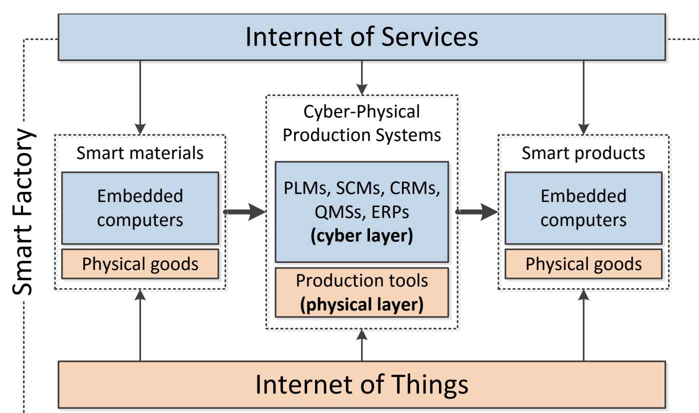


Fig. 2. Fundamental concepts of Industry 4.0

continuous time in accordance to the laws of physics. According to Drath et al. [45], in addition to the physical and cyber components, each CPS requires the a set of services to allow the communication of collected data between the CPS and other actors in the process. As the concept of CPS is inherently broad by its definition and may be used to describe systems in a variety of processes, in the rest of the report we will use the term CPS to denote a system used in the manufacturing process. The manufacturing CPSs are sometimes called Cyber-Physical Production Systems (CPPS). In such CPSs, all physical devices in a manufacturing process are equipped with embedded computers and are connected in a common network. Embedded computers monitor the state of their physical counterparts and create and maintain their virtual representations. Factory monitoring and control systems use the virtual representation and related data streams to manage the manufacturing process.

- **Internet of Things (IoT)** is a paradigm that covers "*the pervasive presence around us of a variety of things or objects - such as Radio-Frequency IDentification (RFID) tags, sensors, actuators, mobile phones, etc. - which, through unique addressing schemas, are able to interact with each other and cooperate with their neighbors to reach common goals*" [16]. IoT can be seen as the direct enabler of CPSs as it provides a foundation for their connecting and networking. This allows CPSs to cooperate through unique addressing schemas and exchange data essential to the manufacturing process.
- **Internet of Services (IoS)** is a vision in which companies communicate with their users and collaborators through the Internet in the form of web services. According to [30], IoS consists of the participants, an infrastructure for services, business models, and web services. Services are offered and combined into value-added groups based on their use by external actors. In the context of Industry 4.0, IoS allows communication of processed and analyzed data collected from CPSs to other interested parties, both internal and external to a manufacturing process.
- **Smart Factory** is defined as "*a factory that context-aware assists people and machines in execution of their tasks*" [95]. The term context-aware system refers to a system that can consider contextual information about an object of interest. Such contextual information may be object position, size, current phase of the production, etc. A smart factory accomplishes its tasks based on the information from both physical world, e. g., machine or product position, and cyber world, e. g., electronic documents, drawings, or simulation models.

Finally, after a short overview of fundamental Industry 4.0 concepts, we present the Industry 4.0 definition given in [68]: "***Industry 4.0 is a collective term for technologies and concepts of value chain organization. Within the modular structured Smart Factories of Industry 4.0, CPSs monitor physical processes, create a virtual copy of the physical world, and make decentralized decisions. Over the IoT, CPSs communicate and cooperate with each other and humans in real time. Via the IoS,***

both internal and cross-organizational services are offered and utilized by participants of the value chain.".

1.2 Automation and Integration in Industry 4.0

The common goal of all innovations that caused industrial revolutions was to increase the automation of the manufacturing process in order to increase the quality and speed of the production. Starting from complex CIMs in 1980's to current trends of using highly connected smart components, automation has been one of the main motivators of industrial development. However, the automation process has changed over the past few years, especially with the emergence of visions such is Industry 4.0.

The automation components are traditionally classified using an "automation pyramid", presented on the left hand side of Figure 3. Bottom three levels are mostly related to the hardware infrastructure, where simple autonomous control actions are performed, e. g., changing temperature or flow, together with various monitoring, performance assessment, and diagnosis functionality. At the plant management level, i. e., with the Manufacturing Execution System (MES), advanced production control algorithms are executed. Further, maintenance management, inventory control, production scheduling operations, and quality assurance may be controlled at this level. At the level of Enterprise Resource Planning (ERP) system, most of the strategic and business related planning is done. An entire supply chain of a company comprising material procurement, manufacturing, storage, transportation, and sales among other processes, is coordinated through an ERP. Since each level requires data from the level below in order to provide services and information to end users, integration between levels is an important issue that needs to be addressed. Therefore, to ensure that a company is operational across all levels, uninterrupted information flow must be provided by the means of device and information system integration. In general, integration in the area of software and system development can be defined as: "***the***

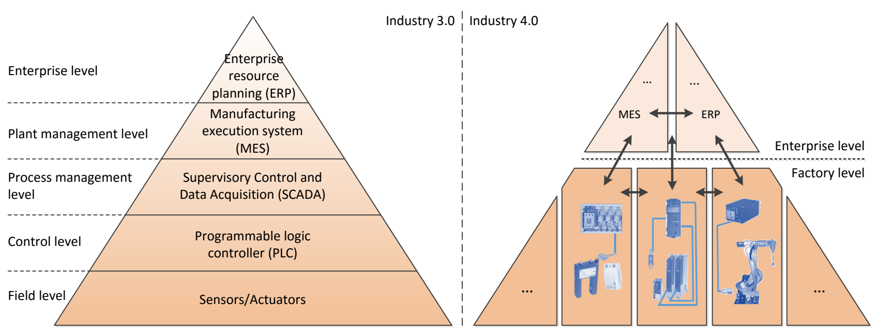


Fig. 3. Automation pyramid

process of linking separate computing systems into a whole so that these elements can work together effectively" [94].

Often, in contemporary manufacturing systems, integration is addressed by the standardization of communication interfaces or by manual development of integration adapters [66]. Although the standardization is the best method for solving the integration issues, device or system manufacturers often adapt standards to suit their own needs or even disregard it and use the proprietary protocol due to the business or technological reasons. Therefore, factory engineers often need to create their own integration adapters that are able to communicate and transform information between the automation pyramid layers. This is a time-consuming, error-prone, costly, and a tedious task overall. Although many standards currently exist, the integration problem still remains unsolved and is one of the major problems and cost-driving factors in the industry [126].

If we take a look at the automation pyramid today, at the right hand side of Figure 3, we can still identify the border between hardware and software oriented layers. At a higher abstraction level all of the layers still exist and the elements of the system can still be classified according to the affiliation to one of these layers. However, if we consider the communication aspect of these systems, the clear borders between the layers have disappeared. As everything is connected inside a smart factory, large amounts of data are exchanged in real time. Majority of materials, devices, and products are now equipped with computing devices (e. g., embedded computers and RFIDs) and networked together regardless of their computing power and purpose. The automation pyramid layer borders have disappeared as many actors in the manufacturing process have become smarter, context-aware, and can send more data to more actors than ever before. For example, a smart product may send data to a manufacturing machine to inform it how to pick it up, where to drill a hole, in which color to paint it, etc. At the same time, the product may inform an MES in which production phase it is currently in. Further, it may inform an ERP about the geographical location of the product in order to update storage quotas and count products. As a lot of new companies enter the market with their own devices, many different protocols and data formats are used for device communication. Now, more than ever, comprehensive horizontal and vertical integration of machines and business application systems is required while implementing a smart factory. Machines on the lowest level have to be vendor-independent, flexible, and efficiently integrated with application systems from the IT-level and possibly with new cloud services. Therefore, a large number of adapters needs to be created in order for the whole system to function as a whole.

1.3 Motivation

With increasing automation and the degree of component coupling, the factors of adaptability, quality, and efficiency of the machine integration play a central role in building and running a smart factory. Currently, the exchange of data within the automation pyramid does not meet future requirements in terms of flexibility and adaptability. As shown schematically in the right part of Figure 3, there is a horizontal gap between the machines at the factory level and the overlying applications and services at the enterprise level. Additionally, there exists a vertical gap between machines from different manufacturers, customers, and domains.

Manufacturers of application systems are facing a challenge to integrate their products into the existing machine landscapes of their customers. Often, the machine and equipment landscape is heterogeneous and characterized by many different interfaces. Despite a variety of standardized industry protocols or exchange standards, machine interfaces are often adapted for a certain domain, manufacturer, or machine. Thus, integration between machines and overlying application systems causes manual adaptation effort which is complex, time-consuming, and expensive. Even in the traditional industry (Industry 3.0) around 40% of the enterprise budget was spent on the information integration tasks [23]. This number is still valid given the ever-growing number of connected devices that need to be integrated. Moreover, quality and transparency of the integration solution are hindered by manual development of the integration solutions.

All of the interfaces participating in the data exchange process send and receive data formatted according to a set of rules, i. e., data formats. In general terms, these data formats and the appropriate set of tools used for their handling may be considered as *technical spaces (TS)*. According to Bezivin, Kurtev et al. [33, 86] the technical space may be defined as: "***a working context with a set of associated concepts, body of knowledge, tools, required skills, and possibilities***". The term technical space originates from the model-driven software engineering field and the term *modeling space* [28] that describes an application domain in which formal modeling methods are applied in order to derive the solution for an identified problem. In order to broaden the scope of this term to the whole field of computer science, Bezivin et al. used an inherently broad construct *working context* making the term technical space applicable even in other engineering disciplines. Some examples of computer science TSs include Comma Separated Values (CSV) TS, eXtensible Markup Language (XML) TS, and Eclipse Modeling Framework (EMF) TS. Examples outside of the computer science field include house construction TS in civil engineering field, where the technical space comprises materials, rules, techniques, and building skills required to construct a proper place for living, and car construction TS in mechanical engineering field with similar TS elements. In this report we consider only computer science TSs with additional Industry 4.0-specific constraints presented in Section 2.

Therefore, to facilitate exchange of data between devices, adapters must be developed for each combination of technical spaces. For example, in order to import data formatted as CSV into an application system that can only read XML, adapters must be developed that transform the data from the CSV TS to the XML TS. In the context of Industry 4.0 where everything is connected and a large number of elements exists in different TSs, manual development of adapters between each pair of communicating devices and systems is a tedious job. This problem of TS disparity in a smart system can be named ***inter-space heterogeneity*** or, according to Wimmer [160], *data model heterogeneity*.

Currently, adapters are usually implemented either by using a programming language specific to the particular combination of TSs (TSL) or by a general purpose language (GPL). TSLs cannot be applied to all possible TS combinations and are seldom used in practice. The benefit of TSLs is the closeness to the integration domain, i. e., they heavily rely on using the concepts of data formats being transformed. Therefore, adapter developers can easily learn and use TSLs which concepts are close to integrated devices. Some examples of the TSLs are eXtensible Stylesheet Language Transformations (XSLT) [35], for specifying transformations in the XML TS, and ATL transformation language (ATL) [76] and Epsilon transformation language (ETL) [83], for specifying transformation in the EMF TS [31]. On the other hand, GPLs such as Java or C# can also be used to integrate any two TSs. These languages often come equipped with libraries that provide parsing ability for data in majority of TSs. However, as these languages are of a general purpose, it is on the developer to create mappings using the parsing libraries with the generic programming language concepts at their disposal. Because of the diversity of data formats and a lack of TSLs, developers nowadays usually opt for GPLs. This further slows down the process of adapter creation, as inappropriate concepts are often used. In the end, none of these languages really provide a reliable and universal solution to the inter-space heterogeneity problem. Therefore, a new approach is needed.

In addition to the inter-space heterogeneity, additional problem of ***intra-space heterogeneity*** is often encountered. According to Wimmer [160], this problem can also be named *structural heterogeneity*. Even if the two devices are integrated with an adapter, the schema according to which a device sends data, may vary based on many factors including device configuration, device version, or the process in which it is used. Intra-space heterogeneity problem introduces even more complexity to the manual implementation of the adapters as they must be robust enough to adapt themselves to encountered changes. Alternatively, manual changes of code are needed in order for the adapter to work properly under new circumstances. This issue can be addressed with the creation of highly reusable and easily adjustable adapters. However, in the world of GPLs this is very hard to accomplish. Although the adapter code can be structured so as to allow easier extension and reuse, the constructs in GPLs are still too generic and not suitable for domain knowledge representation and its reuse.

The existence of the aforementioned heterogeneity issues greatly slows down the development of adapters and indirectly may hinder the performance of the entire manufacturing process. Existing approaches depend to much on the programing languages that are at the inappropriate level of abstraction and usually limited to predefined set of scenarios. Therefore, the aim of the research proposed in this report is to develop an integration language and appropriate integration approach that will alleviate both heterogeneity problems that were previously identified.

Apart from Introduction and Conclusion, the report is organized in three sections. In Section 2 we present a description of the proposed research, while in Section 3 we present possible applications of the proposed research results. The overview of the current state-of-the-art solutions and research work is given in Section 4.

The proposed research project will be partially implemented as a part of COST Action IC1404: Multi-Paradigm Modelling for Cyber-Physical Systems (MPM4CPS) [1]. Although our focus is just on a small part of the CPS domain, mainly on integration of devices and information systems, we feel that the integration is one of the main enablers of future CPS-based systems. To be more specific, the goals and activities of the proposed research align best with goals and activities of the MPM4CPS Work Group 2 (WG2). The compatibility can be seen as follows:

– WG2 - Activity 1: "*Investigate current standards and best practices (modelling languages, interfaces for interoperability, processes, ...) used in CPS*", is already addressed in this initial report in Section 4.
– WG2 - Activity 2: "*Survey state-of-the art on MPM tools and techniques used in different disciplines for CPS development including an efficiency evaluation of MPM tools and techniques on CPS*", is one of the main tasks to be performed as a part of the research project presented in this report. The initial specification of such a tool survey can be seen in Subsection 2.3 and Subsection 4.5.
– WG2 - Activity 3: "*Investigate requirements for future MPM4CPS modelling tools and techniques*" will be performed as a first step in our endeavor to create a "smart" CPS integration approach (technique) and the appropriate tooling support (tool).

2 Description of the Research

In this section we propose a research aimed at specifying an approach to technical space (TS) integration with a main goal to mitigate heterogeneity problems that currently exist in the integration domain. Although there are many possibilities and methodologies to choose from, for the specification of the integration approach we plan to follow Model-Driven Software Development (MDSD) principles. MDSD-based approaches are usually centered around a language that is specific to a certain domain of application (Domain Specific Language, DSL). In this research we are focusing on the domain of TS integration. Several well known benefits of MDSD-based and DSL-centric approaches are: (i) better expressiveness of the approach in the given domain which directly leads to a significant increase in productivity [80], (ii) the approach can be learned and used easier by users from the domain [84], and (iii) the approach would offer a possibility for analysis, verification, optimization, parallelization, and transformation in the terms of domain specific constructs [109].

[1] `http://mpm4cps.eu/`

The notions of MDSD and DSLs and their main characteristics are introduced in the Subsection 2.1, while in Subsection 2.2 we provide further argumentation on why the MDSD is a viable choice when developing an integration approach between TSs. In Subsection 2.2 we also present the goals, hypotheses, and expected results of the proposed research. We provide an overview of the research methodology in Subsection 2.3.

2.1 Research Topic and Basic Terminology

The main topic of the proposed research is the creation of a framework for the integration of TSs based on the main principles of the MDSD approach.

In MDSD, models are considered as the first class entities and they are central artifacts of the software development process. Models are mentally created by the means of mental mapping and reduction in which real world entities are identified, grouped together, generalized, and stripped of properties that are irrelevant for a particular use case. Once a mental model is created, *modeling languages* are needed to provide the appropriate notation for representing these models. Often, developers of a modeling language create the notation in such a way as to provide domain experts, i. e., modelers, visually and semantically familiar concepts from the domain being modeled. Development of a modeling language requires identification of domain concepts that will be mapped to appropriate language concepts and provided a visual or textual notation.

Modeling languages are developed and used according to the MDSD four-level conjecture [15, 32]. The levels are defined according to the degree of abstraction they imply, starting from the lowest. At the bottom level (L0), the real world system exists in which the observed physical entities reside. A *model* of the system is created by the means of an appropriate modeling language, and it resides at the next level (L1). Model contains a virtual representation of the observed system entities with only relevant information about each particular entity. The creation of a modeling language requires that the real system is observed, but instead of focusing on each system entity, different types of entities are identified together with the necessary properties and relationships between them. Specification of entity types, properties, and relationships is called a *meta-model* which resides at the next level (L2) of MDSD conjecture. To create a model, concrete entities are instantiated according to a type, and properties are populated with values. Therefore, a model can be considered as an instance of a meta-model or in another words, a model *conforms to* a meta-model. Meta-models also need to be specified by using a language often refereed to as a *meta-modeling language*. The concepts of such a language do not depend on a particular domain and are defined by the environment in which the meta-models are specified. Meta-modeling language concepts are given in a form of a *meta-meta-model*, which resides at the top level (L3) of the MDSD conjecture. Therefore, each meta-model must conform to a particular meta-meta-model. As there is no practical benefit in introducing new levels of abstraction above L3, meta-meta-models usually conform to themselves and are specified reflectively using their own concepts.

Modeling languages that rely on the domain knowledge and provide concepts close to the target domain are called Domain Specific Modeling Languages (DSML) and can be seen as a subset of a wider term: Domain Specific Languages (DSL) [149]. The advantage of DSMLs in comparison to general purpose modeling languages (GPMLs), such is the Unified Modeling Language (UML) [137], is the closeness to the domain under observation and appropriateness of modeling concepts that are used for the concrete modeling task. By using such a language, a domain expert or a user familiar with the domain is able to specify the solution faster, with less errors, using familiar concepts than it is the case with GPMLs.

Once specified, no model can continue to exist unchanged or isolated. Therefore, operations on models aim at changing them and bridging differences between models. Such operations are called *model transformations*. Analogously to Wirth's well known equation [161] designed for general purpose programming languages "Algorithms + Data Structures = Programs", the following equation may be applied in MDSD approaches "Models + Model Transformations = Software" [28]. Model transformations are specified at the level of meta-models but are executed at the model level. This way, a transformation may be specified once, for a combination of source and target meta-models, and then executed multiple times for each source model that conforms to the source meta-models. The output of such a transformation is a model that conforms to a target meta-model. Model transformations are specified by using a transformation language which can also be characterized as a DSL for the domain of model transformations. Therefore the same language development rules apply as for the development of DSMLs.

As the MDSD approach to model integration relies on model transformations which are based on the four-level conjecture, integrated TSs have to be represented in a suitable way. TSs only deal with the virtual representation of real world entities and as such most of them may be considered to have three levels presented at the right hand side of Figure 4. Together with the real world system (L0), TSs form the appropriate four-level structure in order for them to be a subject of an MDSD integration approach. At the L1, each TS has a model that represents data of the system entities. Such data conforms to a data schema that corresponds to the meta-model notion. Data schemas are specified

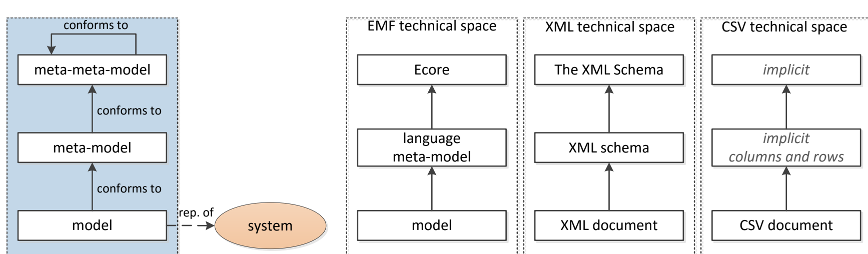


Fig. 4. Three-level TS architecture with examples

with a schema language which concepts form a meta-meta-model. Both meta-models and meta-meta-models can be implicitly or explicitly defined in a TS. Each three-level TS can be seen as based on a single meta-meta-model and a collection of meta-models [31]. Examples of the frequently used three-level TSs are presented at the left hand side of Figure 4. In the proposed research we will focus only on three-level TSs.

2.2 Research Goals and Expected Results

A language for the integration of TSs may be categorized as a DSML for the integration domain. Data originating from the source TS represent a model of the real world device that has sent it. The integration adapters are created at the level of data schemas, i. e., at the level of meta-models, and as such they can be considered as model transformations. Considering all of the aforementioned, we formulate the basic hypothesis of our research:

Hypothesis 0 *It is possible to solve heterogeneity problems in TS integration by creating appropriate DSMLs and following the principles of MDSD approach.*

The main goal of the proposed research, derived directly from this hypothesis, is to define a methodological approach and a software solution in which the MDSD principles and DSMLs will be used to overcome heterogeneity issues in order to allow integration of TSs. The derived hypotheses, which lead to the formulation of research approaches whose aim is to corroborate the Hypothesis 0 are given in the rest of this subsection.

In order for the proposed DSML to be useful in the real world and be able to overcome heterogeneity issues presented in Subsection 1.2, it must satisfy the following two requirements:

1. provide means to integrate two arbitrary TSs with the language concepts that can be easily understood by users familiar with the TSs being integrated, and
2. provide concepts that are reusable and allow for the process of reuse to be automated as much as possible.

The first requirement addresses the problem of inter-space heterogeneity. If the users of such a language understand both the data schema concepts and have a language specifically tailored for the integration domain, they would create adapters easier, faster, and with less effort. Such a language should be understandable by domain experts from any TS domain, just like it is the case with the XML and EMF experts and the integration languages specific to each of these TSs (e. g., XSLT, ATL, ETL, etc.). The development process could be improved even further if the same language would be used for the combination of arbitrary TSs just as GPLs are used. Therefore, such a language must have the benefits of both kinds of languages in order to replace them for the TS integration.

In order to create an integration language that is used across various domains and TSs, different data schemas (i. e., meta-models) must be represented in the same way to be used by the language. There are two possible approaches to creating such a representation. First approach comprises developing one or more DSMLs for each of the TSs. Different DSMLs would enable different type of users to model the same system from different viewpoints. Using the developed DSMLs, users can specify data schemas at a higher abstraction level using concepts close to their comprehension of the domain. The benefit of such approach would be better definition of integration semantics as it is more obvious what concepts from TSs are integrated. However, such approach requires a lot of effort to implement a DSML for each TS, or to adapt existing DSMLs to allow for integration language to be used on top of them. Another drawback of such approach is that a right level of DSML abstraction is hard to achieve. If the abstraction is too high, specified transformation would not have all the necessary information in order to be executed on the data level. If the abstraction is too low, the integration language does not differ much from the data schema already present in the technical space.

The second approach is closer to the system implementation and is based on representing existing TS meta-models with a common representation that is at the same level of abstraction as the original meta-model. As each meta-model comprises entity types, relationships, and properties, it may be possible to find a common, graph-like representation to which all of meta-models from different TSs could be mapped onto. Such a generic representation of TS meta-models would allow for the same integration language to be used for any combination of TSs. Additionally, as the integration adapters must perform transformations on the original source model, such a generic representation must preserve links to the original data elements that will be used in the integration process. Therefore, the following hypothesis may be introduced:

Hypothesis 1 *It is possible to represent data schemas (i. e., meta-models) from the three-level technical spaces in a uniform way by using a graph-like representation, while preserving links to original elements.*

Once the generic representation is provided and the appropriate tools for importing TS meta-models are created, a domain specific integration language may be developed. As it is used to specify relationships between source and target TS in a graphical way, the integration language may be classified as a relationship-based mapping language. Relationship-based mapping systems rely on the specification of high-level relationships between elements (i.e., attributes or sets of attributes) of the source and target TSs. The user starts the mapping design process by providing, usually through a graphical interface, all known attribute correspondences between elements of a source and a target TS. Once such a specification is created, it can be used as an input to other processes such is the generation of adapters and verification of correspondences [10]. Therefore, the next hypothesis of our work is:

Hypothesis 2 *It is possible to create a relationship-based mapping language that allows the creation of high-level mappings between the uniform data schema representations, from which the data integration adapters can be generated.*

The second requirement for the integration language, the reuse of language concepts, mainly addresses the problem of intra-space heterogeneity. The integration language and its concepts should be created in such a way to be easily and automatically reused in new integration projects. Reuse also helps in overcoming the inter-space heterogeneity as integration of new technical spaces could be done on the basis of constructs from previously defined adapters. Although both heterogeneity issues are tackled by implementing a reuse framework and it introduces more complexity to the development of a mapping language, it should be possible to achieve greater degree of reuse automation as the industrial context often comprises similar scenarios slightly adapted to some configuration changes. Therefore, the next hypothesis may be introduced:

Hypothesis 3 *It is possible to create an extensible reuse framework based on the created domain specific integration language that will allow reuse of previously defined integration adapters in the presence of intra-space heterogeneity.*

After introducing these hypotheses, we may also state that the main goal of this research is to provide an MDSD approach for a structured, automated, and reusable integration of TSs. The central idea of the approach is a TS independent coupling component that, in addition to the domain specific integration language, also allows a systematic reuse of integration knowledge from previous integration projects. The reuse or adaptation of existing integration knowledge to new projects is to be provided via framework in an automated and transparent way.

The expected results comprises the following contributions:

- *Theoretical contributions* in the field of model-driven integration of technical spaces. Such contributions will include:
  - survey on existing integration approaches and software solutions;
  - application of MDSD in the TS integration domain relying on a generic representation of meta-model structure;
  - identification of main concepts needed for the implementation of a domain specific language for the integration of TSs;
  - conceptualization of an extendible reuse framework specifically tailored to an Industry 4.0 integration domain;
  - specification of a methodological approach for the application of the developed integration framework; and
  - measurement framework that measures the effort needed to implement an integration adapter in our approach.
- *Development contribution* in the form of a TS integration tool that will implement the MDSD integration approach comprising an integration language and a reuse framework.

– *Application contribution* that comprises application of the integration approach on several use cases and dissemination of evaluation results and lessons learned.

The main expected result of this research is easier and simpler integration of TSs with the aim to improve the response time to production process changes and solve both inter-space and intra-space heterogeneity issues. Expected end-users are integration experts and developers from companies that provide hardware and software solutions for smart factories who need to integrate their products into an existing product landscape. Further, as the term technical space is inherently broad, the results of our research could be used by developers who want to provide data interchange between software which data is structured in a form of a three-level TS. This will be evaluated on the practical use cases that are presented in Section 3.

2.3 Research Methodology

In this subsection we present the following methodological steps of our research: (i) study existing integration software, (ii) develop the integration approach, and (iii) evaluate the integration approach.

Step 1: Study existing integration software During our initial state-of-the-art literature study, presented in this report, we have identified various integration software presented in Subsection 4.5. Identified software solutions constituted an initial set of software to be studied. After the initial set was identified, we eliminated solutions that did not fulfill the criteria of being recently updated, currently used, and available for download. After the literature study, we have searched the Internet for phrases: “integration tool”, “schema matching tool”, “schema mapping tool”, “migration tool”, “ectl tool”, and “bridging tool”. This search resulted in several industrial software solutions that were not identified by our literature study. This also allowed us to classify the solutions by relevance (closeness the integration domain and number of search hits) and choose the most relevant ones for our study. This lead to the omission of large number of solutions, especially in the area of Extract, Clean, Transform, Load (ECTL) processes as ECTL is a very generic notion covering wide range of different processes. Most of the omitted software solutions were used in a narrow domain making them marginally important to our general integration approach.

The next activity that needs to be preformed as a part of the proposed research is the study of these software solutions. Benefits of such a study are twofold. On the one hand side, as we plan to preform the study prior to the development of our integration approach, we can identify advantages and disadvantages of each solution, good practices, and usage patterns. Further, we may identify main concepts that these solutions use to implement integration adapters and to allow knowledge reuse. We will use the obtained experience and information in the development of our approach. Another benefit of such a study is that it represents a baseline for the evaluation of our approach. Once we develop the approach and the appropriate tooling support, we may compare its concepts, performance, and user experience with the solutions that were studied as a part of this methodology step.

The process of evaluating the integration software will be based on a single example which will be implemented in all identified solutions. We will use the example of the integration between sensors, which send CSV data, and information systems, that can receive XML data. The reason of choosing this example is that it is one of our target use cases in which we want to apply our approach. The more detailed description of this use case is given in Subsection 3.2.

While implementing the example in each of the tools, a set of the measured characteristics must be defined in advance so as to perform the comparison and draw usable conclusions. Main characteristics that will be recorded are:

- number and type of supported technical spaces,
- possibility of adding a new technical space,
- description of mapping language characteristics,
- description of expression language characteristics,
- description of knowledge repository characteristics,
- reuse granularity and reusable concepts,
- description of executable code generation mechanisms,
- extension possibility of mapping and transformation mechanisms,
- type of the main application domain,
- software license type, and
- software recentness.

Step 2: Develop the integration approach Development of the integration approach will be performed as an iterative process with the following main activities: requirement solicitation, design, construction, testing, debugging, deployment, and maintenance.

The requirement will be acquired in three ways: (i) interview with experts in the integration domain, (ii) literature study, and (iii) integration software study. The most important source of functional and non-functional requirement are domain experts. As they are also the end users of the tool, their expectations, previous experiences, and lessons learned are of utmost importance for the whole development process. Further requirements will be a direct result of the state-of-the-art literature and integration software study. These requirements cover the current trends and practices in the field of system and data integration.

Based on the defined requirements, the following system elements are to be designed: (i) the integration approach and its phases, (ii) concepts of mapping and expression languages, (iii) reuse algorithm and its steps, and (iv) architecture of supporting tools. After the design phase, construction, testing, and debugging will be performed in order to produce a working integration tool from the specifications provided in the previous development step. The Java language will be used as it provides widely used frameworks for creating DSMLs (EMF [143], Xtext [51], Sirius [152]). Further, using these frameworks will allow us to deploy the tools as a set of plug-ins in the widely-used Eclipse environment [73].

The maintenance activity is beyond the scope of this research and it is planned for the follow up phases.

Step 3: Evaluate the approach The approach will be evaluated on the predefined set of examples and the result of the evaluation will be put into the context of software study results from the Step 1 of this methodology. By applying the approach and appropriate tools on a predefined set of examples, we will test their functionality and domain coverage. This will provide us with the information whether the tool has all the appropriate concepts and functions to be considered for future practical use. Comparison with other tools will allow us to test the efficiency of our approach and the tool in the context of the existing integration software landscape.

The evaluation will be performed on two case studies: (i) integration of the sensor and information system in a smart factory and (ii) exchange of models between different meta-modeling environments. In Section 3 we present these examples in more details.

Before the evaluation, case studies must be prepared. In order for the evaluation to provide relevant information, real world data will be collected, anonymized, and turned into example data set. Further, in order for the tool performance to be protected from external influences, it will be installed on a clean operating system and tested in a controlled environment.

The evaluation will be performed by several integration domain experts in a controlled environment (identical premisses, computers, and the tool setup) with the same amount of given time. After the evaluation is preformed we will disseminate its results and the lessons learned. This will pave the way for future improvements and research directions.

3 Applicability of Research Results

As the proposed approach is based on three-level TSs that are used in a large number of integration use cases, a degree of its practical applicability is high. The approach can be used in the industrial context as a building block of the factory automation. We will focus on a particular problem encountered in every factory: the problem of integration between sensor machines and information systems (ISs). Although we choose one example and focus on it, the same conclusions will be valid for the integration between various sensor machines and also between different information systems in the industrial context.

In addition to this industrial application, the approach is applicable to many other, non-industrial software integration domains. One of the notable problems is the interchange of models and meta-models between meta-modeling environments. This is a known issue as discussed by Kern et al. in [81, 82]. Although meta-modeling environments have the export and import mechanisms, they usually focus on a small number of serialization formats. This hinders the usability of these tools in practice as migration of models and collaboration within a team are made difficult. We will use our approach to provide these environments with an external model interchange functionality. The same approach could be also applied to many other, non-industrial, use cases. For example, ECTL processes in the domain of databases could be specified using our approach. As these processes aim at gathering data from various data sources formatted in different ways, an ECTL process could be seen as the integration of these various TSs from data sources on one side and relational TSs of a relational database on the other. In addition to ECTL processes, a notable application would be in the ontology alignment process in which the alignments can be specified using a mapping language.

In the rest of the section, we introduce two use cases on which we plan to evaluate our approach.

3.1 Model Interchange Between Meta-Modeling Environments

Models play an important role in Domain-Specific Modeling [80] and other related development disciplines. Generally, models represent a system in an abstract way, improve the understanding of a system, and facilitate the communication between different stakeholders. The creation of models is the result of a modeling process which is supported by a modeling tool. A special class of these modeling tools are meta-modeling tools. In addition to providing a user with a set of predefined modeling languages, meta-modeling tools provide a mechanism for the specification of new modeling languages. Examples of meta-modeling tools are: MetaEdit+ [80], Eclipse Modeling Framework [143], and Microsoft Visio [67].

An important requirement for modeling tools, including meta-modeling tools, is the interoperability with other tools. In the context of this use case, interoperability is defined as the ability of two or more tools to exchange models or meta-models. Additionally, these exchanged models and meta-models must be usable in the tools they are imported in. Often, tools support a specific task in the development process. Therefore, a successful application of the whole development process depends heavily on the degree of interoperability between the tools used in the process. Besides the cooperation of tools, the evolution of a tool landscape is an important aspect. As the software industry constantly evolves, modeling tools also evolve and the old ones are being replaced by new tools that better fit customer's needs. In order to avoid the vendor lock-in effect, interoperability between tools is necessary and enables the reuse of existing models between tools from different vendors.

Currently, the interoperability between meta-modeling tools is not widely supported [81, 82]. There is no suitable model exchange approach that takes meta-models into consideration. We will address this lack of interoperability between meta-modeling tools and use the proposed approach to provide the exchange of models in consideration of their meta-models. This way we plan to allow an efficient and user-oriented import and export of models in tools currently used in the industry. In our evaluation, we will focus on the interchange between the two popular environments: Visio [67] and MetaEdit+ [80].

3.2 Data Interchange Between Sensor Machines and Information Systems

Although the approach could be use in a wide range of machine-to-machine, machine-to-IS, and IS-to-IS integration scenarios, we choose a very specific example in order to show all characteristics and implementation details of our approach. This use case, presented in Figure 5, concerns measuring thickness of wafers during their production. This measurement is important to ensure the quality throughout the production process. For this purpose, the measurement (sensor) machines offer different methods, such as, grid, profile, or spot measurements. Depending on the selected method, the machine produces different output data. In this case, each machine produces one CSV file per operation containing measured values. For data processing and analysis, CSV data must be imported into an MES. The MES offers data interfaces which allow the import of XML documents conforming to a defined schema. Beside the inter-space (technical) heterogeneity between the CSV format of the source system and XML format of the target system, the import mechanism must overcome the intra-space (functional) heterogeneity between source and target systems. The existence of different measurement methods lead to a variability in CSV files. Therefore, an MES vendor needs a set of different adapters for the integration of the measuring machines. The manual implementation is in most cases insufficient, time-consuming, costly, and error-prone. Hence, we will use our integration approach in order to develop an integration solution which solves both heterogeneity problems.

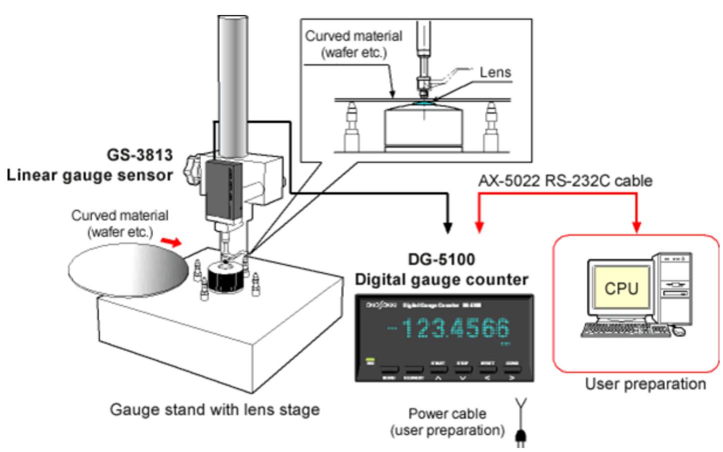


Fig. 5. Integration of a wafer measurement device and an information system

4 State-of-the-Art

In this section we present an overview of state-of-the-art literature in the integration and schema consolidation domains. In general, we can distinguish between the following two mechanisms: standardization and transformation. Standardization can be defined as a development process of a standard which avoids heterogeneity a priori by defining a common structure. For the integration in the context of Industry 4.0 there is a variety of standards which overcome both inter-space and intra-space heterogeneity. Some of the important and novel standards in the age of Industry 4.0 are presented in Subsection 4.1. However, in practice, such standards are frequently adapted to a specific domain, manufacturer, or machine, where a standardized structure of exchanged data is changed. Thus, a mapping or transformation approach is necessary to overcome the heterogeneity between different structures. The aforementioned unification mechanisms are not mutually exclusive. A proprietary structure can be mapped to a standard one by using a transformation.

Traditionally, transformation-based integration approaches were classified under the names Schema matching, Schema mapping, Ontology matching, and Ontology alignment. Schema-based approaches from these categories are presented in Subsection 4.2. As our approach concerns the model-driven integration of technical spaces, we have surveyed existing literature on the topics of model-driven schema matching and model-driven integration of both industrial and non-industrial software systems. This specific subset of schema matching approaches is presented in Subsection 4.3. Ontology-based approaches are presented in Subsection 4.4, while in Subsection 4.5, various commercial and non-commercial solutions in the area of data and system integration are presented with the detailed survey and comparison yet to be performed as a part of the research proposed in this report.

4.1 Industry automation reference models and standards

Most of the integration problems can be solved by introducing standards that are usually made for a particular layer of the manufacturing system. These system layers and standards are often organized in a form of reference architectures to allow easier classification and separation of concerns. In this subsection, we only present several contemporary standards and reference architectures closely related to the proposed research and Industry 4.0 vision.

The Reference Architecture Model for Industry 4.0 (RAMI 4.0) [165], presented in Figure 6, is developed by several German institutions. It illustrates the connection between Information Technology (IT), manufacturers, and products life cycle through a three-dimensional space in which each dimension represents a layered view on this concepts. The left horizontal axis represents the life cycle of facilities and products. Furthermore, a distinction is made between "types" and "instances". A "type" becomes an "instance" when design and prototyping have been completed and the actual product is being manufactured. The main building block according to such a view is an *i4.0 component* representing a

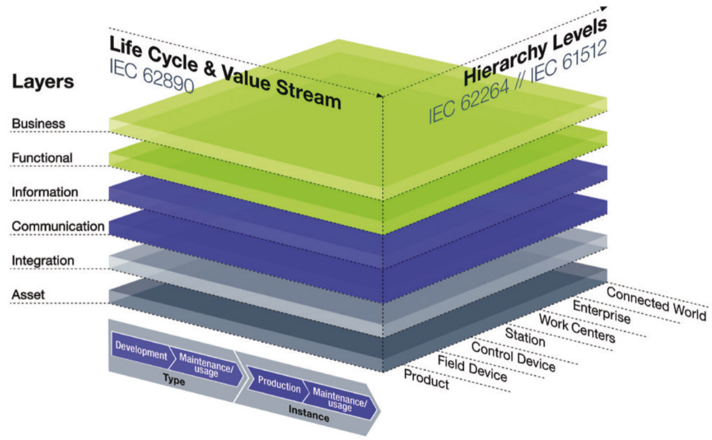


Fig. 6. The Reference Architecture Model for Industry 4.0 (RAMI 4.0), source [165]

unified description of assets, products, and networking information. A management shell should be implemented for each of the i4.0 component assets (e. g., sensor, actuator and plant assets) and stored in a data warehouse. The management shell may be seen as a virtual representation off the real asset containing both status information and data produced by the i4.0 component. The reference model allows the representation of data in the form of management shells during the entire life cycle. Along with the right hand horizontal axis the location of the functionality and responsibilities are given in the hierarchical organization. The reference model broadens the traditional hierarchical levels by adding the Product level at the bottom, and the Connected World that goes beyond the boundaries of the individual factory at the top. In addition, RAMI 4.0 allows the description and implementation of highly flexible concepts. This leverages the transition process of current manufacturing systems to Industry 4.0 by providing an easy step by step migration environment. Left vertical axis represents IT perspective which is comprised of various layers such as business, functional, information, etc. These layers corresponds to the IT way of thinking where complex projects are decomposed into smaller manageable parts.

Complementary to the proposed reference model, several standards are currently used in different organizational and technological layers of an enterprise. As the focus of the future research will be the integration of machines and ISs in the context of Industry 4.0, we describe the following two standards that target similar issue: Automation Markup Language (AutomationML, AML) [46] and Open Platform Communications Unified Architecture (OPC UA) [103].

AML is developed for the field of production systems engineering and commissioning. The data exchange format proposed for AML is an XML schema-based data format developed in order to support the data exchange in a heterogeneous engineering tools landscape. The goal of AML is to allow interconnection of engineering tools from different disciplines, e. g., mechanical plant engineering, electrical design, process engineering, process control engineering, HMI development, PLC programming, robot programming, etc. AML stores engineering information which structure follows the object oriented paradigm and allows modeling of physical and logical plant components as data objects encapsulating different aspects. Typical objects in plant automation comprise information on topology, geometry, kinematics, and logic, whereas logic comprises sequencing, behavior, and control. Therefore, an important focus is on the exchange of object oriented data structures, geometry, kinematics, and logic. AML combines existing industry data formats that are designed for storing and exchanging different aspects of engineering information. These data formats are used on an "as-is" basis within their own specifications and are not branched for AML needs [59].

OPC UA is applicable to manufacturing software in application areas such as Field Devices, Control Systems, Manufacturing Execution Systems, and Enterprise Resource Planning Systems. These systems are intended to exchange information and to use command and control for industrial processes. OPC UA defines a common infrastructure model to facilitate this information exchange. OPC UA specifies: (i) the information model to represent structure, behavior and semantics, (ii) the message model to interact between applications, (iii) the communication model to transfer the data between end-points, and (iv) the conformance model to guarantee interoperability between systems. OPC UA is a platform-independent standard through which various kinds of systems and devices can communicate by sending messages over various types of networks. It supports robust, secure communication that assures the identity of actors in the process and resists attacks. Information is conveyed using OPC UA-defined and vendor-defined data types. OPC UA can be mapped onto a variety of communication protocols and data can be encoded in various ways to trade off portability and efficiency. The OPC UA specifications are layered to isolate the core design from the underlying computing technology and network transport. This allows OPC UA to be mapped to future technologies as necessary, without negating the basic design. Data can be encoded in the form of XML document or an UA Binary representation [61].

The creation of these standards aims to provide a detailed description of the appropriate component of the manufacturing process. As production processes evolve, these standards must grow accordingly to reflect introduced changes. This makes the interoperability between the different standards or even different versions of the same standards a problem to solve.

4.2 Schema-based integration approaches

The proposed research aims at integrating three-level TSs at the meta-model level. Traditionally such approaches were named schema matching approaches [20]. Although the schema matching approaches originate from the domain of relational databases and XML systems, many of the algorithms, approaches, and principles are still valid in general purpose integration.

In the book [20], edited by Bellahsene et al., a survey on schema matching techniques and approaches may be found. This survey focuses on the usage of semantic matching to perform schema evolution and schema merging. It also gives an overview on the currently used matching approaches, visualization, versioning, and collaboration techniques. In this book, the term schema matching is defined as: "***the task of finding semantic correspondences between elements of two schemas***". Although closely related, this term should not be confused with the term schema mapping. According to Ten Cate et al. [146], schema mapping may be defined as "***a high-level, declarative specification of the relationship between two database schemas, typically called the source schema and the target schema.***". Schema mappings are usually specified with a visual notation. Therefore, schema matching systems are not to be confused with the schema mapping systems, where the former one is concerned with (semi-)automatically providing a set of mapping elements but the latter comprises a tool that allows specification of mappings between source and target schemas where the mappings are taken as an input for executable code generators. Schema mapping systems often allow the manual specification of the mappings bat may also contain schema matching modules that can (semi-)automatically assist users in finding the appropriate mapping candidates.

In the recent years, a number of additional surveys and evaluations of schema matching and schema mapping approaches were conducted. Rahm, Bernstein, Do et al. [40, 129] classify existing research work on schema matching by the type of the implemented matching approach. Shvaiko and Euzenat build upon these surveys in [141] and introduce more detailed classification. Based on their type, the following approaches to automatic schema matching are identified in these surveys: (i) *instance-level similarity approaches* that use instance data and identify patterns and other characteristics of the data in order to find matching schema elements, (ii) *single element data or semantic similarity approaches* that use isolated schema element information to find matches, (iii) *element structure similarity approaches* that find matches by comparing schema structures and structural patterns, (iv) *constraint similarity approaches* that compare constraints explicitly or implicitly defined over a set of schema elements, (v) *repository based approaches* that use the previously defined matches and apply it to a new context, (vi) hybrid approaches that directly combine several matching approaches to determine match candidates based on multiple criteria or information sources, and (vii) *composite approaches* that combine the results of several independently executed matchers, including hybrid matchers.

Another contribution of these surveys [40, 129] is the introduction of taxonomy of matching features with the aim to identify possible techniques for automation of the matching process. Rahm and Bernstein propose that a generic *Matcher* tool should have at least: (i) schema importers that convert schemas to a generic representation, (ii) generic match implementation language for the specification of matches between source and target schema elements, and (iii) a global repository for storing identified matches. Although the paper focuses on the web service and database integration domains, conclusions and identified concepts can be generalized and applied to other integration problems as well. The following approaches and tools are classified, described, and compared in detail: Learning Source Descriptions (LSD) [43], Semantic Knowledge Articulation Tool (SKAT) [116], DIKE [120], ARTEMIS [34], Cupid [102], Clio [113], Similarity flooding (SF) [107], Delta [36], Tess [92], Tree matching [158], Autoplex [22], Automatch [21], COMA [41], Embley et al. approach [56], GLUE [44], S-Match [62] and TransSCM [114]. While most of the aforementioned tools aim to solve a matching problem in a specific domain, a few approaches like Clio, Cupid, COMA, and SF, try to address the schema matching problem in a generic way that is suitable for various application domains. Some of the these tools will be evaluated as a part of the tool study proposed in Subsection 4.5, while other tools are either not maintained anymore or are still in the prototype phase.

In their subsequent survey, Bernstein, Rahm, et al. [24] cover the ten years of research and advancement in the field of schema matching that have passed from their initial survey [129]. They present the new emerging approaches to matching elements: (i) *graph matching* that compare schema structures by using graph-based algorithms, (ii) *usage-based matching* that analyzes tool logs for user matching activities, (iii) *document content similarity* that groups instance data into documents and match them based on information retrieval techniques, and (iv) *document link similarity* where concepts in two ontologies are regarded as similar if the entities referring to those concepts are similar. In addition to these information-based techniques, Bernstein et al. also identify plethora of new techniques for creating hybrid and composite approaches with a note that a trend can be spotted of switching from pure machine learning approaches to ontology alignment and matching. Several new tools have also been covered with the survey: COMA++ [17], ASMOV [75], Falcon-AO [70], RiMON [93], AgreementMaker [37], and OII Harmony [139]. The authors argue that due to the availability of large numbers of schemas on the web, a holistic matching approach is becoming quite appealing and it is needed more than ever before. However, the existing approaches have been applied in the domains where the schemas are small, with just a few, well-understood underlying concepts. Therefore, there is a need for a better and practically usable approach as little of the matching technology has made it into commercial offerings.

Based on the findings of their surveys and evaluations of schema matching approaches, Do, Rahm, et al. [17, 42] developed a schema matching tool named COMA++. The tool focuses on the integration of large and complex XML schemas. By a notion of large and complex XML schema, the authors consider schemas with more than 100 schema elements with user types and complex structures defined in it. COMA++ uses composite matchers, combining the power of simple matchers into one that is usually more efficient or more suitable for a specific application domain. The tool follows a divide and conquer approach, where the schema is modularized and modules are mapped independently. Afterward, a full mapping is created by merging the individual module mappings.

Unlike aforementioned surveys on general matching techniques, a survey focusing on XML schema matching is provided by Agreste et al. [2]. The authors significantly extend the scope of published surveys with a description of new techniques particularly tailored for the XML domain. Agreste et al. argue that in order to have a best fit matching technique in the domain of XML, the matching tools should be specialized for that domain and use all of its peculiarities. This way, the matches are found more efficiently, matches are more appropriate to the domain, and the greatest advantage is that the schema element semantics can be identified in a more precise way. They also provide a template, called *XML Matcher Template*, which proposes the main components and their roles and behaviors in any XML matcher. Agreste et al. also discuss several commercial prototypes designed to identify mappings between XML schemas. These prototypes are then classified by using the degree of correspondence to their XML Matcher template.

We have also identified several schema matching approaches not covered by the aforementioned surveys. At the Faculty of Technical Sciences, University of Novi Sad, a tool named Integrated Information Systems Studio (IIS*Studio) is developed with one of its core function being the integration and consolidation of relational database schemas and subschemas. The main purpose of IIS*Studio is information system development which comprises the conceptual database schema design, based on the *form type* concept [55, 98], and development of appropriate business applications. IIS*Studio comprises three main tools: IIS*Case [97, 101], IIS*UIModeler [19], and IIS*Ree [5]. IIS*Case is the core tool of IIS*Studio and provides the following functionality:

1. conceptual modeling of database schemas, transaction programs, and business applications of an IS [98, 122–124],
2. specification of check constraint at the level of a conceptual model [118],
3. automated design of relational database subschemas in the 3rd normal form (3NF) [96, 99],
4. automated integration of subschemas into a unified database schema in the 3NF [96, 99, 100, 131–133],
5. automated generation of SQL/DDL code for various database management systems (DBMSs) [4], and
6. automated generation of executable prototypes of business applications.

In the case of large systems being developed by the incremental approach, a system is decomposed into several subsystems that are modeled independently and usually by different designers. The process of independent design of subsystems and their database schemas may lead to collisions in expressing the real world constraints and business rules. Therefore, in IIS*Case, the process of system integration is not just a mere unifying of its subsystems. It is based on detecting and resolving all the formal constraint collisions. Luković, Ristić et al. [96, 99, 100, 131–133] proved that, at the level of relational data model, it is possible to automatically detect formal collisions of database constraints embedded into different subschemas, where each subschema represents a database schema of a sole IS subsystem. If collisions are detected, at least one subschema is formally not consistent with the current version of a database schema of a whole system. Programs made over inconsistent subschemas do not guarantee logically correct database updates. Therefore, the authors created and embedded into IIS*Case algorithms for detecting formal constraint collisions for the most often used constraint types at the level of relational data model. Besides, they embedded into IIS*Case a number of collision reports that assist designers in their resolving. By this, the database schema integration process based on the approach of a gradual integration of subschemas into a unified database schema is supported by IIS*Case in a large extent.

At the abstraction level of platform independent models (PIMs), IIS*UIModeler provides conceptual modeling of common user interface (UI) models, as well as business applications that include specifications of: (i) UI, (ii) structures of transaction programs aimed to execute over a database, and (iii) basic application functionality that includes the following "standard" data operations: read, insert, update, and delete. A PIM of business applications is combined with a selected common UI model and then automatically transformed into the program code. In this way, fully executable application prototypes are generated. IIS*Ree is a model-driven re-engineering tool that provides a set of extractors and model-to-model transformations that extract and transform relational database schemas to a conceptual model based on the form type concept. Once the conceptual model is adapted to new requirements, a set of new model-to-model transformations and code generators is used to generate relational database schema and deployment scripts.

Bernstein et al. [25] introduce a solution that aims to bring the schema mapping technique to an industrial environment. They present a prototype of a customizable schema matcher called PROTOtype PLAtform for Schema Matching (PROTOPLASM). PROTOPLASM comprises three layers: (i) an import layer in which the mapped artifacts are transformed into a common internal representation based on XML, (ii) operation layer which comprises concepts needed to build a schema matching strategy, and (iii) a graphical language layer in which the graphical representations of operational concepts are combined into matching strategy scripts which are then executed. Similarly, Raghavan et al. [128] propose a solution, named SchemaMapper, which uses a hyperbolic tree instead of a linear tree representation. In their opinion, the hyperbolic tree contributes to a faster human-performed search for an element that is needed for a matching process. Another difference between PROTOPLASM and SchemaMapper is that the latter uses a tabular mapping representation instead of line-based one, which is traditionally used. While the line-based representation may lead to overcrowded diagrams in the case of large schemas, tabular representation leads to more compact views. A drawback of the SchemaMapper is reflected in the fact that it is focused only on the XML technical space.

Alexe et al. [6, 7, 10] propose an approach to schema mapping in the domain of relational database schema integration. Unlike most of the previously listed solutions, that load entire source and target schemas and create high-level mappings between them, Alexe's approach named "divide-design-merge" allows splitting source and target schemas into smaller parts, creating mappings between these parts, and merging all partial mappings into a whole as the final step. This approach has been supported by three tools that authors have developed. Eirene [13] is a schema mapping design tool that takes as an input a set of data examples provided by the user. In turn, Eirene outputs a schema mapping that "fits" the set of data examples, if such a schema mapping exists. Afterward, a user can interact with the Muse [8] tool to refine and further design schema mappings through the use of data examples. Finally, in the merge phase, a global schema mapping is generated through the correlation of the individual schema mappings by using a MapMerge [9] application.

Muse is one of the earliest systems that adopted a different approach to schema-mapping design. This approach uses instance data examples to infer mappings between schemas according to which these data are formatted. In these approaches, schema matching does not rely on a high-level schema mapping language, but on algorithms that analyze instance data to find data constraints or patterns which are often very good indicators of the similarity between the appropriate schema elements. This type of an approach to schema mapping has been also proposed by [7, 12, 38, 63, 146, 163]

In [48], Duchateau and Bellahsene present Yet Another Matcher (YAM). YAM is a self-tuning and extensible matcher factory tool that generates a best-fit schema matching algorithm for a specific integration scenario. Based on the generated matching algorithm schema element matches are then identified and proposed to a user. The *self-tuning feature* of this approach provides the ability to produce a matcher with appropriate, user-defined, characteristics for a given scenario. The *extensible feature* enables users of a matching tool to add new similarity measures and thus increase the overall effectiveness of the system. The goal of YAM is to alleviate users of a manual configuration of matcher similarity measures including the thresholds setup and iterative adjustment of these measures. YAM automatically tunes these parameters by relying on the implemented machine learning techniques. Similar techniques were implemented in MatchPlanner [49], which is based on the decision tree while, and eTuner [91] that performs the same job by employing a set of synthetic matching scenarios involving the schema being mapped. For each eTuner synthetic scenario correct matches are known in advance and thus it is possible to evaluate produced mapping configurations.

In addition to approaches and tools described in research papers, several patents have been filed concerning schema matching approaches, notations, and systems. Thomas [147] patented a schema matching system based on a tabular representation of schemas and mapping formulas. The proposed system displays instance data beside the appropriate schema elements in order to give the user better contextual understanding of the schema elements. Once the schemas are loaded and represented in a tabular layout, textual formulas can be specified to represent relations between source and target elements. In her second patent, Thomas [148] introduces the notion of a platform independent schema representation, named *conceptual model* which is a high level representation of schema understandable to a domain expert. The reminder of the patent is similar to the one presented in [147]. In [140], Seligman patents a semi-automatic schema matching approach based on a linguistic processing of schema elements. Element relations, i. e., matches, are discovered by analyzing element names with a machine learning algorithm that uses both generic and domain thesauri together with the list of frequently used abbreviations. Match probabilities are provided to a user who manually chooses the mappings he deems a best-fit. Patents [69,135], filed by Hobbs and Robertson et al. respectively, propose notations and layout algorithms to be used in matching tools. Both patents propose that mappings are represented as lines with a central (algorithmic) part of the mapping being shaped as a box to allow easier handling and spotting. Hobbs also proposes an algorithm that handles drawing and layout of the mappings used while users create mappings, load previous work, or scroll the schema elements in their views.

4.3 Model-driven integration approaches

The Model-Driven Software Development (MDSD) promotes the development of software systems at different levels of abstraction, and Domain Specific Languages (DSLs) play a prominent role to reduce development costs. As one of the most time-consuming and error-prone parts of introducing a new technology or a new functionality to the existing IT landscape is integration, by means of an appropriate DSL software engineers can design a software system that can later be integrated and deployed to a variety of specific platforms using automatic transformations. As the transformations are specified at the level of meta-model, i. e., data schema, transformation rules may be seen as schema matching rules. Therefore, MDSD transformation approaches may be seen as a subset of schema matching and mapping approaches.

Büttner et al. [29] present a model-driven approach to the data integration between government institutions in Germany. The integration approach is centered around the standardization of messages, interfaces, and models of data that are being exchanged. A compliance with the standards is regulated by a central governing body that governs the specification of meta-models, i. e., data formats, for different sectors in the German government. As different standards exist, integration is essential task that needs to be performed in order for the data to be exchanged. Therefore, integration processes need to be used at th meta-model level to allow transformation of messages and their communication to other German or European institutions. Büttner et al. have developed a central repository, named XRepository, that stores all meta-modeling concepts, well-formedness rules, and process and semantic specifications that together form standards. The XGenerator tool is used to produce artifacts that are used in the integration process. These artifacts are usually web service specifications that need to be implemented by software vendors to integrate their solutions with the system.

Agt et al. [3], Kutsche et al. [87, 88], and Milanović at al. [111] present a meta-modeling approach to the integration of heterogeneous distributed IT systems named BIZYCLE. The BIZYCLE integration process is based on multilevel modeling abstractions. The integration scenario is first modeled at the computation independent level, where business aspects of an integration scenario are described. The model is then refined at the platform specific level, where technical interfaces of the systems that should be integrated are described. For each of the supported platforms: SAP, relational and XML databases, web services, XML files, J2EE components, and .NET applications, a specific platform specific model is created. The automation of the integration process is achieved through model extraction, systematic conflict analysis process, and code generation. Reuse is supported at the model-level via BIZYCLE Repository [112], as interface descriptions, transformation rules, and semantic annotations can be stored and shared between projects and users.

Wimmer [160] developed a meta-model bridging framework and a graphical DSL that provides bridging of different technical spaces based on data mining techniques. The framework comprises a mapping view and a transformation view. At the mapping view level, a user defines mappings between elements of two meta-models using the provided DSL. Thereby a mapping expresses also a relationship between model elements, i. e., instances of meta-models. In Wimmer's approach, mappings between meta-model elements are defined with mapping operators which are considered as processing entities encapsulating a certain kind of transformation logic. A set of applied mapping operators, also called a mapping model, defines the mapping from a left hand side (LHS) meta-model to a right hand side (RHS) meta-model. Thus, the mapping model declaratively describes the semantic correspondences on a high-level of abstraction. The transformation view is capable of executing the defined mapping models. During the execution, a mapping operator takes as input elements of the source model and produces as output semantically equivalent elements of the target model.

Huh et al [71] developed Marama Torua, a tool supporting high-level specification and implementation of complex mappings of data schemas. Complex mapping relationships are represented in multiple notational forms and users are provided with a semi-automated mapping assistance for large models. Multiple views are implemented in order to ease the process of mapping specifications for all levels of source and target schema complexity. The tool supports creation of mappings between any two technical spaces. However, if the import tool has not been already developed for a certain technical space, a user must develop it manually and map a data schema to a generic tool structure. Marama Torua comprises a set of Eclipse plug-ins allowing close integration with other tools such as schema browsers.

There are several DSLs and frameworks that are not directly related to the schema mapping, but fit better to the fields of schema matching and enterprise application integration. Vuković et al. [153, 154] present a language called Semantic-Aided Integration Language (SAIL). This language allows for the matching components to be described, generated, and used in their framework without having to be implemented in a general purpose programming language and are available without having to rebuild the entire application. The aim of the developed matching framework is to automate some of the steps in conflict resolution of the matching process. Interfaces and their elements can be semantically described using ontologies in order to facilitate this automation. Although the approach itself is based on the ontology alignment principles, the SAIL domain specific language is used to specify matching algorithms and follows all the principles of the MDSD methodology.

Another domain specific language, named Highway, is developed by Kovanović et al. [85]. Highway is developed as an internal DSL in the Clojure programming language. It may be used for implementing enterprise application integration solutions in a technology independent and functional manner. Highway uses functional programming techniques in order to simplify enterprise application integration development.

Sleiman et al. [142] propose a DSL called Guarana and a software tool to design and automatically deploy integration solutions in order to reduce integration costs. Guarana provides a set of domain specific constructors to design integration solutions. It provides an expressive graphical notation for these constructors, which allows a user to visually design an integration solution. Functions and mappings are all displayed on the same diagram thus giving a good overview of the general solution.

The Federated USer Exchange (FUSE) approach [157] represents a domain-aware approach to user model interoperability. It consists of a manual mapping process and an automatic translation process. Both processes contain two domain aware mechanisms: (i) a canonical user model and (ii) user model mapping transforms, which tailor the processes to specific domains. All mappings are first created with the canonical user model as a target. This model represents a consistent shared user model. The user model mapping transforms are mapping components specifically created and used for mapping between different user models via the canonical model. This approach differs from existing generic approaches because it incorporates domain knowledge in new processes and tools to support complex user model interoperability tasks in multiple overlapping domains.

Wischenbart et al. [162] employed a MDSD approach to the integration of data collected from social networks. Although many social networks on the Web allow access via dedicated APIs, the extraction of instance data for further use by applications is often a tedious task. As a result, instance data transformation to Linked Data in the form of Ontology Web Language (OWL), as well as the integration with other data sources are needed. This paper proposes a model-driven approach to overcome data model heterogeneity by automatically transforming schemas and instance data from JavaScript Object Notation (JSON) to OWL/XML. Authors specify a set of transformations that transform an input model, i. e., the model of JSON messages collected from social network APIs, to the OWL model, i. e., ontology used for representing social network data instances and their semantic.

In addition to aforementioned approaches, model-to-model (M2M) transformation languages can be also seen as possible means to integrate different TSs.

M2M transformation languages are specified at the level of a meta-model but are executed at the model level. Therefore it is required to extract or use existing meta-models from the integrated TSs. Once meta-models are obtained, transformation rules may be specified with one of the transformation languages. A definition and a classification of M2M transformation languages is given by Mens et al. [108] and the overview of the selected visual transformation languages may be found in our previous paper [39]. The advantage of these languages is that they are supported out of the box with well defined notations and semantics, they are usually declarative, and as we deal with the three layered technical spaces, meta-models already exist or can be easily extracted to a desired environment. However, the disadvantage of these languages is that they may be seen as general purpose mapping languages, and they are not well suited for the domain of integration thus the transformations may be verbose and hard to read and maintain.

4.4 Ontology-based integration approaches

What is known as schema mapping or schema matching in database and artificial intelligence domains, in semantic web community it is known under the name Ontology Alignment [52] or Ontology Matching [57]. The task of these approaches is to find groups of elements sharing the same semantics. Majority of the tools presented in Subsection 4.2 can be also applied to the ontology alignment process. Therefore, there is no clear line that separates these approaches and fit them into a single category. For example, although both approaches described in [153, 162] may be seen primarily as MDSD approaches, they rely on the use of ontology alignment techniques and principles to find best match candidates.

Unlike schema matching approaches that usually comprise techniques for guessing the meaning encoded in the data schemas, ontology matching systems try to exploit knowledge explicitly encoded in ontologies. In their survey [141], Shvaiko et al. focus on the comparison of the following ontology alignment solutions: Naive Ontology Mapping (NOM) [54], Quick Ontology Mapping (QOM) [53], OWL Lite Aligner (OLA) [58], Anchor-PROMPT [117]. Many other ontology alignment solutions are compared in a survey by Ardjani et al. [14]. The ontology alignment is performed according to a strategy or a combination of techniques for calculating similarity measures by using a set of parameters, e. g., weighting parameters and thresholds, and a set of external resources, e. g., thesauri and dictionaries. As a result, a set of semantic links between ontology entities is obtained. In addition to the tool comparison, Ardjani et al. introduce a classification of the ontology approaches based on the similarity measurement methods that are used. The identified methods are as follows: (i) *terminological methods* that are using terms, strings, and text for comparison, (ii) *structural methods*, which calculate the similarity by exploiting structural information, (iii) *extensional methods*, which infer the similarity between two entities, especially concepts or classes, by analyzing their extensions, i.e. their instances, and (iv) *semantic methods*, which include the methods based on an external thesauri and dictionaries and on deductive techniques that heavily rely on logical models, such as propositional satisfiability or description logic.

4.5 Integration Tools

In addition to identifying the state-of-the-art research work on the topics of schema matching, mapping, and integration, our goal is also to identify visual mapping software solutions that are currently used in practice. As the goal of our research is to develop a visual mapping language for the integration domain, a survey on such tools will provide us with valuable information on current integration trends, best practices, and characteristics of an industry-ready solution.

We have identified a number of software solutions that follow similar approach to the one we propose as a part of the future research. All the solutions may be categorized into three groups based on their dominant application domain:

- *General mapping tools*: Altova MapForce [104], AnalytiX Mapping Manager [47], Open Mapping Software [134], Vorto [151], Karma [65], Talend Enterprise Data Integration Studio [27], CLIP [127], Schema Mapper [128], Mint Mapping Tool [115], MetaDapper Data Mapping Tool [136], ++Spicy [105], Coma++ [17, 42], and Schema Mapper Transformer [138].
- *XML mapping tools*: Liquid XML Studio (DataMapper component) [145] and Stylus Studio (Data Direct XML converter i XQuery Mapper) [144].
- *ECTL tools*: Adeptia ETL Tool [1], Informatica ETL Suite [74], Oracle Data Integrator [119], OpenRefine (also known as GoogleRefine) [150], Datastage ETL Tool [72], Microsoft SQL Server Integration Services (SSIS) [110], and Clover ETL Tool [121].

According to the research plan given in Section 2.3, the first step of the future research will be to implement a single example in all of the identified software solutions. By doing this, we will gather a set of tool characteristics and compare them. The result of such a comparison will be disseminated in the future research reports. Such a comparison will also provide us with a possibility to evaluate our tool by comparing its characteristics with the characteristics of the identified software.

To our knowledge there are no existing studies that evaluate and compare integration software based on the proposed criteria, taking care of their domain coverage, language complexity, supported functions, reuse ability, availability, and other non-technical characteristics. In [130], Rathinasamy compares only several tools for the domain of asset management. These tools include COMA++ and Altova MapForce. Do et al. [40] compare several matching tools based on the matching algorithm performance. However, these tools are either still in the prototype phase or not maintained any more, which makes the proposed survey even more relevant.

A framework aimed at generating test cases for visual mapping system evaluation was developed by Alexe et al. [11]. The framework generates test cases that are used to test if a mapping tool language covers a set of predefined functionality, i. e., basic mapping scenarios, such as copying a value from a source to a target model, constant value generation, horizontal and vertical partitioning, etc. Most of these mapping scenarios come from the domain of database schema integration. A full list and a detailed description of the proposed mapping scenarios may be found in [11].

5 Conclusion

In this report we present issues of inter-space and intra-space heterogeneity that often occur during the integration of different technical spaces in the context of Industry 4.0. These issues hinder the productivity and efficiency of a manufacturing system as they often require custom integration adapters to be developed manually. Such a development process is often time-consuming, error-prone, and costly.

In order to solve these issues, we propose a research with goal to provide a model-driven approach to the technical space integration. As technical spaces often have a three-level architecture (data, data schema, and meta-schema levels), we feel that such a problem is well suited for a model-driven integration approach as it is based on the same thee-level conjecture. The proposed research comprises several main phases and should result in the following elements:

- a model-driven approach to technical space integration with the formally defined phases and responsibilities;
- a formally defined and developed mapping language that is used for a specification of integration adapters in a model-driven way;
- a reuse module to improve the speed of adapter development by reusing knowledge from previous projects; and
- an appropriate tooling support for the approach.

We believe that a model-driven approach and appropriate tools will allow integration experts to overcome the heterogeneity issues in a more efficient and productive way. To evaluate this statement, we specify two main practical use cases: (i) integration of machines and information systems in the context of a smart factory and (ii) integration of (meta-modeling) environments as to provide interoperability mechanism. Use cases are chosen as to provide a diverse and more precise evaluation of the approach.

During the state-of-the-art literature survey we have found a plethora of tools and approaches which purpose is to integrate technical spaces. Since a significant number of these research papers are published in the last five years we can conclude that the field of integration and schema mapping is active and growing. As our main focus is on the model-driven techniques in technical space integration, we have invested a lot of effort to find approaches that use this methodology. Although several model-driven integration solutions exist, they are mostly focused on a narrow domain of integration, such as database or XML schema integration. We have not been able to find practically applicable solutions that could be efficiently used to overcome the heterogeneity issues.

Due to the practical need and lack of similar solutions, the presented approach to model-driven integration of technical spaces deserves the attention in the form of the proposed research.

References

1. Adeptia: Adeptia ETL tool (2016), `https://adeptia.com/solutions/ETL-software-for-data-transformation.html`
2. Agreste, S., De Meo, P., Ferrara, E., Ursino, D.: XML matchers: approaches and challenges. Knowledge-Based Systems 66, 190–209 (2014)
3. Agt, H., Bauhoff, G., Cartsburg, M., Kumpe, D., Kutsche, R., Milanović, N.: Metamodeling foundation for software and data integration. In: Information Systems: Modeling, Development, and Integration, Lecture Notes in Business Information Processing, vol. 20, pp. 328–339. Springer, Berlin (2009)
4. Aleksić, S.: An SQL Generator of Database Schema Implementation Scripts in IIS*Case Tool. Master Thesis, University of Novi Sad, Novi Sad (2006)
5. Aleksić, S.: Methods of Database Schema Transformations in Support of the Information System Reengineering Process. PhD thesis, University of Novi Sad, Novi Sad (2013)
6. Alexe, B.: Interactive and Modular Design of Schema Mappings. Ph.D. thesis, University of California at Santa Cruz, Santa Cruz, CA, USA (2011)
7. Alexe, B., Cate, B.T., Kolaitis, P.G., Tan, W.C.: Characterizing schema mappings via data examples. ACM Transactions on Database Systems (TODS) 36(4), 23 (2011)
8. Alexe, B., Chiticariu, L., Miller, R.J., Tan, W.C.: Muse: Mapping understanding and design by example. In: Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on. pp. 10–19. IEEE (2008)
9. Alexe, B., Hernández, M., Popa, L., Tan, W.C.: MapMerge: correlating independent schema mappings. The VLDB Journal 21(2), 191–211 (Apr 2012)
10. Alexe, B., Tan, W.C.: A New Framework for Designing Schema Mappings. In: Tannen, V., Wong, L., Libkin, L., Fan, W., Tan, W.C., Fourman, M. (eds.) In Search of Elegance in the Theory and Practice of Computation, pp. 56–88. No. 8000 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2013), dOI: 10.1007/978-3-642-41660-6_4
11. Alexe, B., Tan, W.C., Velegrakis, Y.: STBenchmark: towards a benchmark for mapping systems. Proceedings of the VLDB Endowment 1(1), 230–244 (2008)
12. Alexe, B., Ten Cate, B., Kolaitis, P.G., Tan, W.C.: Designing and refining schema mappings via data examples. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of data. pp. 133–144. ACM (2011)
13. Alexe, B., Ten Cate, B., Kolaitis, P.G., Tan, W.C.: EIRENE: Interactive design and refinement of schema mappings via data examples. Proceedings of the VLDB Endowment 4(12), 1414–1417 (2011)
14. Ardjani, F., Bouchiha, D., Malki, M.: Ontology-Alignment Techniques: Survey and Analysis. International Journal of Modern Education & Computer Science 7(11) (2015)
15. Atkinson, C., Kühne, T.: Model-driven development: a metamodeling foundation. Software, IEEE 20(5), 36–41 (2003)
16. Atzori, L., Iera, A., Morabito, G.: The Internet of Things: A survey. Computer Networks 54(15), 2787–2805 (Oct 2010)

17. Aumueller, D., Do, H.H., Massmann, S., Rahm, E.: Schema and ontology matching with COMA++. In: Proceedings of the 2005 ACM SIGMOD international conference on Management of data. pp. 906–908. ACM (2005)
18. Baheti, R., Gill, H.: Cyber-physical systems. The impact of control technology 12, 161–166 (2011)
19. Banović, J.: An Approach to Generating Executable Information System Specifications. Ph.D. thesis, University of Novi Sad, Novi Sad (2010)
20. Bellahsene, Z., Bonifati, A., Rahm, E., others: Schema matching and mapping, vol. 57. Springer (2011)
21. Bergamaschi, S., Castano, S., Vincini, M., Beneventano, D.: Semantic Integration of Heterogeneous Information Sources. Data Knowl. Eng. 36(3), 215–249 (Mar 2001)
22. Berlin, J., Motro, A.: Autoplex: Automated Discovery of Content for Virtual Databases. In: Batini, C., Giunchiglia, F., Giorgini, P., Mecella, M. (eds.) Cooperative Information Systems, pp. 108–122. No. 2172 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (Sep 2001), dOI: 10.1007/3-540-44751-2_10
23. Bernstein, P.A., Haas, L.M.: Information integration in the enterprise. Communications of the ACM 51(9), 72–79 (2008)
24. Bernstein, P.A., Madhavan, J., Rahm, E.: Generic schema matching, ten years later. Proceedings of the VLDB Endowment 4(11), 695–701 (2011)
25. Bernstein, P.A., Melnik, S., Petropoulos, M., Quix, C.: Industrial-strength schema matching. ACM SIGMOD Record 33(4), 38–43 (2004)
26. Bledowski, K.: The Internet of Things: Industrie 4.0 vs. the Industrial Internet (Jul 2015), `https://www.mapi.net/forecasts-data/internet-things-industrie-40-vs-industrial-internet`
27. Bowen, J.: Getting Started with Talend Open Studio for Data Integration. Packt Publishing Ltd (2012)
28. Brambilla, M., Cabot, J., Wimmer, M.: Model-Driven Software Engineering in Practice. Morgan & Claypool Publishers, San Rafael (2012)
29. Büttner, F., Bartels, U., Hamann, L., Hofrichter, O., Kuhlmann, M., Gogolla, M., Rabe, L., Steimke, F., Rabenstein, Y., Stosiek, A.: Model-driven standardization of public authority data interchange. Science of Computer Programming 89, 162–175 (Sep 2014)
30. Buxmann, P., Hess, T., Ruggaber, R.: Internet of Services. Business & Information Systems Engineering 1(5), 341–342 (Sep 2009)
31. Bézivin, J.: Model driven engineering: An emerging technical space. In: Generative and transformational techniques in software engineering, Lecture Notes in Computer Science, vol. 4143, pp. 36–64. Springer (2006)
32. Bézivin, J., Gerbé, O.: Towards a precise definition of the OMG/MDA framework. In: Automated Software Engineering, 2001.(ASE 2001). Proceedings. 16th Annual International Conference on. pp. 273–280. IEEE (2001)
33. Bézivin, J., Kurtev, I.: Model-based technology integration with the technical space concept. In: Proceedings of the Metainformatics Symposium. Springer-Verlag (2005)
34. Castano, S., Antonellis, V.D.: Global viewing of heterogeneous data sources. IEEE Transactions on Knowledge and Data Engineering 13(2), 277–297 (Mar 2001)
35. Clark, J.: XSL Transformations (XSLT). Recommendation, World Wide Web Consortium (W3C) (1999), `https://www.w3.org/TR/xslt`
36. Clifton, C., Housman, E., Rosenthal, A.: Experience with a combined approach to attribute-matching across heterogeneous databases. In: Data Mining and Reverse Engineering, pp. 428–451. Springer (1998)

37. Cruz, I.F., Antonelli, F.P., Stroe, C.: AgreementMaker: efficient matching for large real-world schemas and ontologies. Proceedings of the VLDB Endowment 2(2), 1586–1589 (2009)
38. Das Sarma, A., Parameswaran, A., Garcia-Molina, H., Widom, J.: Synthesizing view definitions from data. In: Proceedings of the 13th International Conference on Database Theory. pp. 89–103. ACM (2010)
39. Dimitrieski, V., Luković, I., Aleksić, S., Čeliković, M., Milosavljevic, G.: An Overview of Selected Visual M2m Transformation Languages. pp. 450–455. Society for Information Systems and Computer Networks, Kopaonik, Serbia (Mar 2014)
40. Do, H.H., Melnik, S., Rahm, E.: Comparison of schema matching evaluations. In: Web, Web-Services, and Database Systems, pp. 221–237. Springer (2002)
41. Do, H.H., Rahm, E.: COMA: a system for flexible combination of schema matching approaches. In: Proceedings of the 28th international conference on Very Large Data Bases. pp. 610–621. VLDB Endowment (2002)
42. Do, H.H., Rahm, E.: Matching large schemas: Approaches and evaluation. Information Systems 32(6), 857–885 (2007)
43. Doan, A., Domingos, P., Halevy, A.Y.: Reconciling schemas of disparate data sources: A machine-learning approach. In: ACM Sigmod Record. vol. 30, pp. 509–520. ACM (2001)
44. Doan, A., Madhavan, J., Domingos, P., Halevy, A.: Learning to map between ontologies on the semantic web. In: Proceedings of the 11th international conference on World Wide Web. pp. 662–673. ACM (2002)
45. Drath, R., Horch, A.: Industrie 4.0: Hit or Hype? IEEE Industrial Electronics Magazine 8(2), 56–58 (Jun 2014)
46. Drath, R., Lüder, A., Peschke, J., Hundt, L.: AutomationML-the glue for seamless automation engineering. In: Emerging Technologies and Factory Automation, 2008. ETFA 2008. IEEE International Conference on. pp. 616–623. IEEE (2008)
47. DS, A.: Analytix Mapping Manager (2016), http://analytixds.com/amm/
48. Duchateau, F., Bellahsene, Z.: YAM: A Step Forward for Generating a Dedicated Schema Matcher. In: Hameurlain, A., Küng, J., Wagner, R. (eds.) Transactions on Large-Scale Data- and Knowledge-Centered Systems XXV, pp. 150–185. No. 9620 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2016), dOI: 10.1007/978-3-662-49534-6_5
49. Duchateau, F., Bellahsene, Z., Coletta, R.: A flexible approach for planning schema matching algorithms. In: On the Move to Meaningful Internet Systems: OTM 2008, pp. 249–264. Springer (2008)
50. Dujin, A., Geissler, C., Horstkötter, D.: Industry 4.0: The New Industrial Revolution - How Europe Will Succeed. Tech. rep., Roland Berger Strategy Consultants Gmbh, Munich (2014), https://www.rolandberger.com/media/pdf/Roland_Berger_TAB_Industry_4_0_20140403.pdf
51. Efftinge, S., Völter, M.: oAW xText: A framework for textual DSLs. In: Workshop on Modeling Symposium at Eclipse Summit. vol. 32, p. 118 (2006)
52. Ehrig, M.: Ontology Alignment: Bridging the Semantic Gap, Semantic Web And Beyond Computing for Human Experience, vol. 4. Springer (2007)
53. Ehrig, M., Staab, S.: QOM–quick ontology mapping. In: The Semantic Web–ISWC 2004, pp. 683–697. Springer (2004)
54. Ehrig, M., Sure, Y.: Ontology mapping–an integrated approach. In: The Semantic Web: Research and Applications, pp. 76–91. Springer (2004)

55. Čelikovic, M., Luković, I., Aleksić, S., Ivančević, V.: A MOF based meta-model and a concrete DSL syntax of IIS*Case PIM concepts. Computer Science and Information Systems 9(3), 1075–1103 (2012)
56. Embley, D.W., Jackman, D., Xu, L.: Multifaceted Exploitation of Metadata for Attribute Match Discovery in Information Integration. In: Workshop on information integration on the Web. pp. 110–117. Citeseer (2001)
57. Euzenat, J., Shvaiko, P., others: Ontology matching, vol. 18. Springer (2007)
58. Euzenat, J., Valtchev, P., others: Similarity-based ontology alignment in OWL-lite. In: ECAI. vol. 16, p. 333 (2004)
59. e.V., A.: AutomationML Whitepaper. Whitepaper, AutomationML e.V. (Oct 2014), https://www.automationml.org/o.red/uploads/dateien/1417686950-AutomationML%20Whitepaper%20Part%201%20-%20AutomationML%20Architecture%20v2_Oct2014.pdf
60. Evans, P.C., Annunziata, M.: Industrial Internet: Pushing the Boundaries of Minds and Machine. Tech. rep., General Electrics (Nov 2012)
61. Foundation, O.: OPC UA - Overview and Concepts v1.03. Industry Standard Specification, OPC Foundation, Scottsdale, USA (Oct 2015), https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-1-overview-and-concepts/
62. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: S-Match: an algorithm and an implementation of semantic matching. In: ESWS. vol. 3053, pp. 61–75. Springer (2004)
63. Gottlob, G., Senellart, P.: Schema mapping discovery from data instances. Journal of the ACM (JACM) 57(2), 6 (2010)
64. Groover, M.P.: Automation, Production Systems, and Computer-Integrated Manufacturing. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edn. (2007)
65. Gupta, S., Szekely, P., Knoblock, C.A., Goel, A., Taheriyan, M., Muslea, M.: Karma: A system for mapping structured sources into the Semantic Web. In: The Semantic Web: ESWC 2012 Satellite Events, pp. 430–434. Springer (2012)
66. Harjunkoski, I., Nyström, R., Horch, A.: Integration of scheduling and control—Theory or practice? Computers & Chemical Engineering 33(12), 1909–1918 (Dec 2009)
67. Helmers, S.A.: Microsoft Visio 2013 Step By Step. Microsoft Press, Sebastopol, CA, 1 edition edn. (May 2013)
68. Hermann, M., Pentek, T., Otto, B.: Design principles for Industrie 4.0 scenarios: a literature review. Working Paper No. 01-2015, Technische Universität Dortmund, Dortmund (2015)
69. Hobbs, R.L.V.: Mapping for mapping source and target objects (Mar 2009), http://www.google.com/patents/US7499943
70. Hu, W., Qu, Y., Cheng, G.: Matching large ontologies: A divide-and-conquer approach. Data & Knowledge Engineering 67(1), 140–160 (2008)
71. Huh, J., Grundy, J., Hosking, J., Liu, K., Amor, R.: Integrated data mapping for a software meta-tool. In: Software Engineering Conference, 2009. ASWEC'09. Australian. pp. 111–120. IEEE (2009)
72. IBM: InfoSphere DataStage (Jan 2016), http://www-03.ibm.com/software/products/en/ibminfodata
73. Inc, E.F.: Eclipse environment (2016), https://eclipse.org/
74. Informatica: Data Integration Tools and Software Solutions (2016), https://www.informatica.com/products/data-integration.html#fbid=RNZoAPRdLqq

75. Jean-Mary, Y.R., Shironoshita, E.P., Kabuka, M.R.: Ontology matching with semantic verification. Web Semantics: Science, Services and Agents on the World Wide Web 7(3), 235–251 (Sep 2009)
76. Jouault, F., Allilaire, F., Bézivin, J., Kurtev, I.: ATL: A model transformation tool. Science of Computer Programming 72(1-2), 31–39 (Jun 2008)
77. Kagermann, H., Helbig, J., Hellinger, A., Wahlster, W.: Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0. Tech. rep., Forschungsunion (2013)
78. Kagermann, H., Lukas, W.D., Wahlster, W.: Industrie 4.0: Mit dem Internet der Dinge auf dem Weg zur 4. industriellen Revolution. VDI nachrichten 13, 2 (2011)
79. Kalpakjian, S., Schmid, S.: Manufacturing, Engineering and Technology. Digital Designs, 7 edn. (2006)
80. Kelly, S., Tolvanen, J.P.: Domain-specific modeling: enabling full code generation. John Wiley & Sons (2008)
81. Kern, H.: Study of Interoperability between meta-modeling tools. In: 2014 Federated Conference on Computer Science and Information Systems (FedCSIS). pp. 1629–1637 (Sep 2014)
82. Kern, H., Hummel, A., Kühne, S.: Towards a comparative analysis of meta-metamodels. In: Proceedings of the compilation of the co-located workshops on DSM'11, TMC'11, AGERE!'11, AOOPES'11, NEAT'11, & VMIL'11. pp. 7–12. SPLASH '11 Workshops, ACM, New York, NY, USA (2011)
83. Kolovos, D.S., Paige, R.F., Polack, F.A.: The Epsilon Transformation Language. In: Theory and Practice of Model Transformations, pp. 46–60. Springer (2008)
84. Kosar, T., Mernik, M., Carver, J.C.: Program comprehension of domain-specific and general-purpose languages: comparison using a family of experiments. Empirical Software Engineering 17(3), 276–304 (Aug 2011)
85. Kovanovic, V., Djuric, D.: Highway: a domain specific language for enterprise application integration. In: Proceedings of the 5th India Software Engineering Conference. pp. 33–36. ACM (2012)
86. Kurtev, I., Bézivin, J., Akşit, M.: Technological spaces: An initial appraisal. pp. 1–6. Irvine, USA (Nov 2002)
87. Kutsche, R., Milanović, N.: (Meta-)Models, Tools and Infrastructures for Business Application Integration. In: Kaschek, R., Kop, C., Steinberger, C., Fliedl, G. (eds.) Information Systems and e-Business Technologies, pp. 579–584. No. 5 in Lecture Notes in Business Information Processing, Springer Berlin Heidelberg (Apr 2008)
88. Kutsche, R., Milanović, N., Bauhoff, G., Baum, T., Cartsburg, M., Kumpe, D., Widiker, J.: BIZYCLE: Model-based Interoperability Platform for Software and Data Integration. In: Proceedings of the MDTPI at ECMDA. vol. 430 (2008)
89. Lander, E., Holdren, J.P.: Accelerating U.S. Advanced Manufacturing. Report to the president, President's Council of Advisors on Science and Technology, Washington, USA (Oct 2014)
90. Lasi, H., Fettke, P., Kemper, H.G., Feld, T., Hoffmann, M.: Industry 4.0. Business & Information Systems Engineering 6(4), 239–242 (Aug 2014)
91. Lee, Y., Sayyadian, M., Doan, A., Rosenthal, A.S.: eTuner: tuning schema matching software using synthetic scenarios. The VLDB Journal—The International Journal on Very Large Data Bases 16(1), 97–122 (2007)
92. Lerner, B.S.: A model for compound type changes encountered in schema evolution. ACM Transactions on Database Systems (TODS) 25(1), 89–127 (2000)
93. Li, J., Tang, J., Li, Y., Luo, Q.: RiMOM: A Dynamic Multistrategy Ontology Alignment Framework. IEEE Transactions on Knowledge and Data Engineering 21(8), 1218–1232 (Aug 2009)

94. Linthicum, D.S.: Enterprise application integration. Addison-Wesley Professional (2000)
95. Lucke, D., Constantinescu, C., Westkämper, E.: Smart factory-a step towards the next generation of manufacturing. In: Manufacturing systems and technologies for the new frontier, pp. 115–118. Springer (2008)
96. Luković, I.: Integration of Information System Database Module Schemas. Ph.D. thesis, University of Novi Sad, Novi Sad (1995)
97. Luković, I., Aleksić, S., Ivančević, V., Čeliković, M.: DSLs in Action with Model Based Approaches to Information System Development. In: Formal and Practical Aspects of Domain-Specific Languages. IGI Global (Sep 2012)
98. Luković, I., Mogin, P., Pavićević, J., Ristić, S.: An approach to developing complex database schemas using form types. Software: Practice and Experience 37(15), 1621–1656 (Dec 2007)
99. Luković, I., Ristić, S., Mogin, P.: A methodology of a database schema design using the subschemas. In: Proceedings of IEEE International Conference on Computational Cybernetics. Budapest, Hungary (2003)
100. Luković, I., Ristić, S., Mogin, P., Pavićević, J.: Database Schema Integration Process–A Methodology and Aspects of Its Applying. Novi Sad Journal of Mathematics 36(1), 115–140 (2006)
101. Luković, I., Ristić, S., Popović, A., Pavićević, J., Kordić, S., Obrenović, N., Mogin, P., Mostić, J., Govedarica, M.: IIS*Case (2012)
102. Madhavan, J., Bernstein, P.A., Rahm, E.: Generic schema matching with cupid. In: VLDB. vol. 1, pp. 49–58 (2001)
103. Mahnke, W., Leitner, S.H., Damm, M.: OPC unified architecture. Springer Science & Business Media (2009)
104. MapForce, A.: User & Reference Manual. User Manual (2013)
105. Marnette, B., Mecca, G., Papotti, P., Raunich, S., Santoro, D., others: ++ Spicy: an Open-Source Tool for Second-Generation Schema Mapping and Data Exchange. Clio 19, 21 (2011)
106. Mathas, C.: Industry 4.0 is closer than you think (Dec 2013), http://www.edn.com/design/wireless-networking/4425363/Industry-4-0-is-closer-than-you-think
107. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In: Proceedings. 18th International Conference on Data Engineering. pp. 117–128. IEEE (2002)
108. Mens, T., Gorp, P.V.: A Taxonomy of Model Transformation. Electronic Notes in Theoretical Computer Science 152, 125–142 (Mar 2006)
109. Mernik, M., Heering, J., Sloane, A.M.: When and how to develop domain-specific languages. ACM computing surveys (CSUR) 37(4), 316–344 (2005)
110. Microsoft: MicrosoftSQL Server Integration Services (SSIS) (2016), https://msdn.microsoft.com/en-us/library/ms169917.aspx
111. Milanović, N., Cartsburg, M., Kutsche, R., Widiker, J., Kschonsak, F.: Model-Based Interoperability of Heterogeneous Information Systems: An Industrial Case Study. In: Paige, R.F., Hartman, A., Rensink, A. (eds.) Model Driven Architecture - Foundations and Applications, pp. 325–336. No. 5562 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (Jun 2009)
112. Milanović, N., Kutsche, R., Baum, T., Cartsburg, M., Elmasgünes, H., Pohl, M., Widiker, J.: Model&Metamodel, Metadata and Document Repository for Software and Data Integration. In: Czarnecki, K., Ober, I., Bruel, J.M., Uhl, A.,

Völter, M. (eds.) Model Driven Engineering Languages and Systems, pp. 416–430. No. 5301 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (Sep 2008)
113. Miller, R.J., Hernández, M.A., Haas, L.M., Yan, L.L., Ho, C.H., Fagin, R., Popa, L.: The Clio project: managing heterogeneity. SIgMOD Record 30(1), 78–83 (2001)
114. Milo, T., Zohar, S.: Using schema matching to simplify heterogeneous data translation. In: VLDB. vol. 98, pp. 24–27. Citeseer (1998)
115. Mints, C.: Mint - the mapping Tool (2016), `http://mint.image.ece.ntua.gr/redmine/projects/mint/wiki/Mapping_Tool`
116. Mitra, P., Wiederhold, G., Jannink, J.: Semi-automatic integration of knowledge sources (1999)
117. Noy, N.F., Musen, M.A.: Anchor-PROMPT: Using non-local context for semantic matching. In: Proceedings of the workshop on ontologies and information sharing at the international joint conference on artificial intelligence (IJCAI). pp. 63–70 (2001)
118. Obrenović, N.: An Approach to Design, Consolidation and Transformations of Database Schema Check Constraints Based on Platform Independent Models. Ph.D. thesis, University of Novi Sad, Novi Sad (2015)
119. Oracle: Oracle Data Integrator (Jun 2016), `http://www.oracle.com/technetwork/middleware/data-integrator/overview/index.html`
120. Palopoli, L., Terracina, G., Ursino, D.: The System DIKE: Towards the Semi-Automatic Synthesis of Cooperative Information Systems and Data Warehouses. In: ADBIS-DASFAA Symposium. pp. 108–117 (2000)
121. Pavlis, D.: CloverETL Rapid Data Integration (2016), `http://www.cloveretl.com/`
122. Popović, A., Luković, I., Ristić, S.: A Specification of the Structures of Business Applications in the IIS* Case Tool. Info M–Journal of Information Technology and Multimedia Systems 25, 17–24 (2008)
123. Popović, A.: An Approach to Specification of Application System Executable Models. PhD thesis, University of Montenegro, Podgorica (2013)
124. Popović, A., Luković, I., Dimitrieski, V., Djukic, V.: A DSL for modeling application-specific functionalities of business applications. Computer Languages, Systems & Structures 43, 69–95 (2015)
125. Poslad, S.: Ubiquitous Computing: Smart Devices, Environments and Interactions. Wiley, Chichester, U.K, 1 edition edn. (Apr 2009)
126. Pulier, E., Taylor, H.: Understanding enterprise SOA. Manning Greenwich, Conn (2006)
127. Raffio, A., Braga, D., Ceri, S., Papotti, P., Hernandez, M.A.: Clip: a Visual Language for Explicit Schema Mappings. In: 2008 IEEE 24th International Conference on Data Engineering. pp. 30–39 (Apr 2008)
128. Raghavan, A., Rangarajan, D., Shen, R., Gonçalves, M.A., Vemuri, N.S., Fan, W., Fox, E.A.: Schema mapper: a visualization tool for DL integration. In: Digital Libraries, 2005. JCDL'05. Proceedings of the 5th ACM/IEEE-CS Joint Conference on. pp. 414–414. IEEE (2005)
129. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. the VLDB Journal 10(4), 334–350 (2001)
130. Rathinasamy, K.: Comparison of Schema and Data Integration tools for the Asset Management Domain. Ph.D. thesis, MS Thesis, University of South Australia (2011)

131. Ristić, S.: Research on the Problem of Database Subschema Consolidation. Ph.D. thesis, University of Novi Sad, Novi Sad (2002)
132. Ristić, S., Luković, I., Pavičević, J., Mogin, P.: Resolving Database Constraint Collisions Using IIS*Case Tool. Journal of information and organizational sciences 31(1), 187–206 (2007)
133. Ristić, S., Mogin, P., Luković, I.: Specifying database updates using a subschema. In: Proceedings of the 7th IEEE International Conference on Intelligent Engineering Systems. pp. 203–212. Assiut–Luxor, Egypt (2003)
134. Robert Worden: Improving Data Quality with Open Mapping Tools. White paper, Open Mapping Software Ltd (Feb 2011)
135. Robertson, G.G., Churchill, J.E., Czerwinski, M.P., Panditharadhya, P.S., Bhaskara, U.: Schema mapper (Oct 2012), http://www.google.com/patents/US8280923
136. Rozsa, R.: MetaDapper (2016), http://www.metadapper.com/
137. Rumbaugh, J., Jacobson, I., Booch, G.: The Unified Modeling Language Reference Manual. Pearson Higher Education (2004)
138. SchemaMapper: SchemaMapper Transformer (Mar 2016), https://knowledge.safe.com/articles/1136/schemamapper-transformer-tutorial.html
139. Seligman, L., Mork, P., Halevy, A., Smith, K., Carey, M.J., Chen, K., Wolf, C., Madhavan, J., Kannan, A., Burdick, D.: OpenII: an open source information integration toolkit. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. pp. 1057–1060. ACM (2010)
140. Seligman, L.J., Mork, P.D.S., Korb, J.G., Samuel, K.B., Wolf, C.S.: Tools and methods for semi-automatic schema matching (Jan 2008), http://www.google.com/patents/US20080021912
141. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. In: Journal on data semantics IV, pp. 146–171. Springer (2005)
142. Sleiman, H.A., Sultán, A.W., Frantz, R.Z., Corchuelo, R., others: Towards Automatic Code Generation for EAI Solutions using DSL Tools. In: JISBD. pp. 134–145 (2009)
143. Steinberg, D., Budinsky, F., Merks, E., Paternostro, M.: EMF: eclipse modeling framework. Addison-Wesley, Boston, USA, 2 edn. (2008)
144. Studio XML, S.: XML Data Integration, XML Tools, Web Services and XQuery (2016)
145. Technologies, L.: Liquid XML Studio (2016), https://www.liquid-technologies.com/xml-studio
146. Ten Cate, B., Kolaitis, P.G., Tan, W.C.: Schema mappings and data examples. In: Proceedings of the 16th International Conference on Extending Database Technology. pp. 777–780. ACM (2013)
147. Thomas, S.M.: Schema mapping and data transformation on the basis of layout and content (Jul 2012), http://www.google.com/patents/US8234312
148. Thomas, S.M.: Schema mapping and data transformation on the basis of a conceptual model (Dec 2014), http://www.google.com/patents/US8924415
149. Van Deursen, A., Klint, P., Visser, J.: Domain-Specific Languages: An Annotated Bibliography. Sigplan Notices 35(6), 26–36 (2000)
150. Verborgh, R., De Wilde, M.: Using OpenRefine. Packt Publishing Ltd (2013)
151. Vorto, Eclipse: Eclipse Vorto - IoT Toolset for standardized device descriptions (2016), https://www.eclipse.org/vorto/index.html
152. Vujović, V., Maksimović, M., Perišić, B.: Sirius: A rapid development of DSM graphical editor. In: 18th International Conference on Intelligent Engineering Systems (INES). pp. 233–238. IEEE, Tihany, Hungary (2014)

153. Vuković, e., Milanović, N., Bauhoff, G.: Prototype of a Framework for Ontology-aided semantic conflict resolution in enterprise integration. In: Proceedints of 5th International Conference on Information Society and Technology. Society for Information Systems and Computer Networks, Kopaonik, Serbia (Mar 2015)
154. Vuković, e., Milanović, N., Vaderna, R., Dejanović, I., Milosavljević, G.: SAIL: A Domain-Specific Language for Semantic-Aided Automation of Interface Mapping in Enterprise Integration. In: On the Move to Meaningful Internet Systems: OTM 2015 Workshops. pp. 97–106. Springer (2015)
155. Wahlster, W.: Industry 4.0: From Smart Factories to Smart Products (May 2012)
156. Waldner, J.B.: CIM: principles of computer-integrated manufacturing. John Wiley & Sons (1992)
157. Walsh, E., O'Connor, A., Wade, V.: The FUSE domain-aware approach to user model interoperability: A comparative study. In: Information Reuse and Integration (IRI), 2013 IEEE 14th International Conference on. pp. 554–561. IEEE (2013)
158. Wang, J.T.L., Zhang, K., Jeong, K., Shasha, D.: A system for approximate tree matching. Knowledge and Data Engineering, IEEE Transactions on 6(4), 559–571 (1994)
159. Weiser, M.: The computer for the 21st century. Scientific american 265(3), 94–104 (1991)
160. Wimmer, M.: From mining to mapping and roundtrip transformations–a systematic approach to model-based tool integration. Ph.D. thesis, Vienna University of Technology (2008)
161. Wirth, N.: Algorithms + Data Structures = Programs. Prentice Hall, Englewood Cliffs, N.J., 1st edition edn. (Feb 1976)
162. Wischenbart, M., Mitsch, S., Kapsammer, E., Kusel, A., Lechner, S., Pröll, B., Retschitzegger, W., Schönböck, J., Schwinger, W., Wimmer, M.: Automatic data transformation: breaching the walled gardens of social network platforms. In: Proceedings of the Ninth Asia-Pacific Conference on Conceptual Modelling-Volume 143. pp. 89–98. Australian Computer Society, Inc. (2013)
163. Yan, L.L., Miller, R.J., Haas, L.M., Fagin, R.: Data-driven understanding and refinement of schema mappings. In: ACM SIGMOD Record. vol. 30, pp. 485–496. ACM (2001)
164. Zuehlke, D.: SmartFactory—Towards a factory-of-things. Annual Reviews in Control 34(1), 129–138 (2010)
165. ZVEI: The Reference Architectural Model For industrie 4.0 (RAMI 4.0) (2015)

Several Issues on Composition of Cyber-Physical Systems Based on Principles of the Two-Hemisphere Modelling

Oksana Nikiforova[1], Nisrine El Marzouki[1], Nadezda Kunicina[1], Hans Vangheluwe[2], Florin Leon[3], Mauro Iacono[4], Rima Al-Ali[5], and Priscill Orue[6]

[1] Riga Technical University, Latvia
{oksana.nikiforova,nisrine.marzouki,nadezda.kunicina}@rtu.lv
[2] University of Antwerp, Nederlands
hans.vangheluwe@uantwerp.be
[3] Technical University "Gheorghe Asachi" of Iasi, Romania
florin.leon@tuiasi.ro
[4] Seconda Universita di Napoli, Italy
mauro.iacono@unina2.it
[5] Charles University in Prague, Czech Republic
alali@d3s.mff.cuni.cz
[6] University of Malaga, Spain
porue@lcc.uma.es

Abstract. The two-hemisphere model-driven (2HMD) approach assumes modelling and use of procedural and conceptual knowledge on an equal and related basis. This differentiates 2HMD approach from pure procedural, pure conceptual, and object oriented approaches. The approach may be applied in the context of modelling of a particular business domain as well as in the context of modelling the knowledge about the domain. Cyber-physical systems are heterogeneous systems, which requires multidisciplinary approach to their modelling. Modelling of cyber-physical systems by 2HMD approach gives an opportunity to transparently decompose and analyse system's components to be provided and components actually provided, and, thus, to identify and fill the gaps between desirable and actual system content.

Keywords: Two-hemisphere model-driven approach, cyber-physical systems, system composition

1 Introduction

Cyber-Physical Systems (CPS) have never been more central to the corporate strategy today. The features they offer, reliability, performance and robustness are the queen qualities that allow companies to be competitive. To cope with the complexity of the execution of such heterogeneous systems, it is necessary to define an approach to tame its complexity. This approach should be flexible and generic in order to adapt to any type of component of such system and thus, should offer an ability to manage system composition. The Two-hemisphere model driven (2HMD) approach has been successfully applied for domain modelling and software design [1]. One of the most distinguished features of this model is its applicability for both human understanding and automatic transformations [2]. In this paper we illustrate the way how 2HMD approach may be applied to the task of modelling and composition of CPS.

The goal of the paper is to show the way how the problem of complex system composition from smaller parts can be solved by using 2HMD approach for modelling of CPS components. From the point of view of 2HMD approach each component of CPS may be considered as a conceptual class, which preforms the particular operations and meet the defined requirements. The requirements are derived from the model that consists of functional and conceptual "hemispheres". Thus 2HMD approach is applicable for both modelling of components and modelling of the process of to be supported by that component at the same level of abstraction. Moreover, the 2HMD approach can help to identify conflict situations, where the additional analysis is required for sharing responsibilities between system components.

Features of CPS or Cyber-Physical Systems and the necessity to model and decompose them are discussed in Section 2. The essence of the two-hemisphere model driven approach is clarified in Section 3. Application of 2HMD approach for composition and modelling of components of CPS are outlined in Section 4. Conclusions are given in Section 5.

2 Cyber-Physical Systems in the Context of System Composition

A cyber-physical system (CPS) is a mechanism controlled or monitored by computer-based algorithms, tightly integrated with Internet and its users. In CPSs, physical and software components are deeply intertwined, each operating on different spatial and temporal scales, exhibiting multiple and distinct behavioral modalities, and interacting with each other in a myriad of ways that change with context [3], [4].

Being CPS inherently complex, any significant analysis is a challenge. The behavior results from the different scales of the effect of the emerging or fundamental phenomena, the different nature of the components, the interactions and the drawbacks of the internal compositions. A comparable domain is System of systems (SoS), in which analysis exploits decomposition (or, equivalently, design exploits composition), and the system additionally exhibits emerging behaviors or features that have to be modeled at a higher level. While both for CPS or SOS an holistic approach is viable for a general understanding of the system from the point of view of an external observer, in order to design or assess in detail its behaviours the heterogeneity of the problems that have to be analyzed require that every aspect and every component, and every scale and every hierarchical subsystem, need a proper model and a proper modelling technique. This is also a natural consequence of the wide set of expertise that has to be involved in the design, the maintenance and the management of a CPS: every professional takes care of a different aspect of a subsystem, using a specialized view on it that privileges his responsibilities and the modus operandi typical of his field. Building up a comprehensive model of the system could then benefit from the application of an approach that allows model composition and use of different modelling approaches together in a coordinated framework, such as multiformalism [5] or multiparadigm [6] modelling.

For what is related to the design and the analysis of non functional specifications of system components, some frameworks exist that allow multiformalism modelling for non functional specifications by providing, by means of metamodelling, the ability of designing new modelling languages that can be naturally integrated with existing ones and support modularity and multformalism: e.g., the SIMTHESys approach [7] explores these possibilities and is being extended to include also support for a domain, hybrid systems [8], that includes CPS. The main directions behind this approach aim to decouple whenever possible the state spaces of the various components (with significant results in favourable cases [9]), build natural interactions between models written in different modelling languages, and representing emerging properties of the overall model.

Leveraging this experience, it is possible to abstract the main ideas and take inspiration for implementing ideally similar compositional and modular features in modelling approaches that focus on aspects that are different from non functional specification.

In this paper we want to discuss the applicability of some of the concepts here presented about non functional specifications to a more abstract modelling domain, specially focusing of the perspective that allows the application of the 2HMD approach.

CPS involves transdisciplinary approaches, merging theory of cybernetics, mech-tronics, design and process science [10], [11], [12] The process control is often referred to as embedded systems. In embedded systems the emphasis tends to be more on the computational elements, and less on an intense link between the computational and physical elements. CPS is also similar to the Internet of Things (IoT) sharing the same basic architecture, nevertheless, CPS presents a higher combination and coordination between physical and computational elements [13].

As far as CPS are multidisciplinary heterogeneous systems, its implementations requires the strategy of modelling, decomposition into smaller components and their composition into the whole system. The components of such systems then should be detailed at the same level of abstraction.

One of those systems is Ensemble Component-Based Systems [14][15], where components are autonomous and the communication is implicit. Needless to say that emergent systems as an up-and-coming systems introduce new concepts for design [14][16] as well as for whole system development process. Working with such distributed systems require dynamicity and scalability with reserving to the autonomous behavior in each component. This allows designers to have their focus on individual components, and work on developing the links between computational and physical elements for each individual alone.

In those terms, [17][18] introduce approaches to capture internal and external uncertainty in the system, and handle it during adaptation process by linking the physical elements in the same abstraction level as computational ones. In [17], the work presented Ordinary Differential Equations (ODE) for physical objects at the process level. The goal is to capture the impact of delays, which are caused by networks or computational parts, on physical elements (i.e. actuators). The method allows to enhance prediction of the real state boundaries of each physical object. While [18] targets the uncertainty caused by sensor readings, where the precision of sensed data is the main concern. The effect of data precision is represented in self-adaptation process, where the authors extend mode-switch logic to involve statistical testing. The extended logic applies hypothesis testing over historical data to evaluate the condition in mode-switching with a certain confidence level. Worth to mention that mode-switch conditions deal with short time prediction as well as the current situation. At the end, both kind of uncertainty is captured explicitly in the architectural view allowing for applying traditional analysis and transformations with a minimum amount of modification on the existing tools.

We can discuss the issue of decomposition in the context of multiagent systems also. Sometimes, the complex interactions between the individual agents give rise to an emergent behaviour of the system as a whole, e.g. in modelling social systems, traffic simulations, etc. However, other applications can benefit from system decomposition, e.g. agent-based business process modelling. A tool that is often used in this situation is the visual notation of Role-Activity Diagrams [19]. They contain roles, which describe the behaviour of a set of role instances. Roles have states, similarly to dynamic systems. A business process may contain one or more active instances of the same role. An actor is an agent that enacts a role instance. Activities are the basic building blocks of a role. Carrying out the activities of a role can be interpreted as transferring the process control from a state to another state. An activity may be carried out in isolation or may require coordination with activities in other roles, and in this case it is an interaction. Some studies showed that this class of workflows can be formalized and modelled by a concise set of distinct rules in a generic knowledge-based business agent architecture [20] and can be implemented using both agent-oriented languages, such as Jason, and general-purpose functional languages, such as F# [21].

3 The Essence of the Two-Hemisphere Model-Driven Approach

The variety of modelling capabilities and the ability to express links traceability are decisive assets to manage system's complexity. The transformation tool takes one model as input and produces a second model as its output. Two hemisphere model driven or 2HMD approach [1] proposes using of business process modelling and concept modelling to represent systems in the platform independent manner and describes how to transform business process models into UML models, shown in Figure 1.

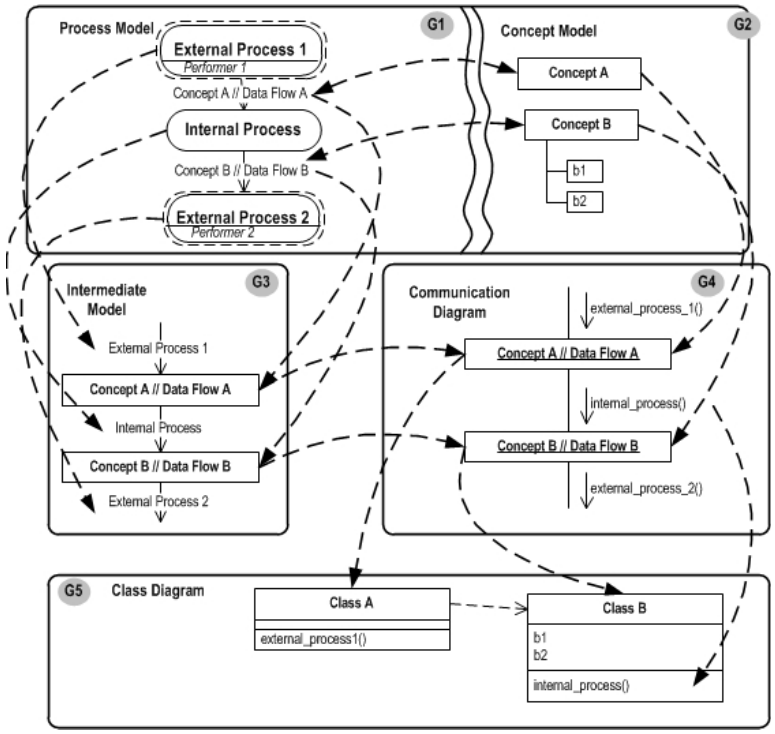


Fig. 1. The essence of the two-hemisphere model driven approach.

Two-hemisphere model consists of two diagrams - business process diagram and conceptual class diagram. The inclusion of these diagrams is not random, it is not only based on the previously mentioned analogy with the human brain, but also based on that information shown in these diagrams helps to describe the system from different points of view, which is important in system development. Business process modelling, as [22] mentioned, developed as a result of solutions made by Management Science and Computer Science in the 1970s. Besides, nowadays the importance of business process modelling has not decreased. The importance of business process modelling is confirmed by [23] regular researches about importance and usability of these processes. Researches confirm that management of business processes is important and companies pays attention to it. This research also shows, that companies over time learn existing business process modelling notations and methodologies. Thereby one advantage of using two-hemisphere model is no need to make additional models to use it, but user can count on, that business model consisting of elements required by two-hemisphere model in the organization already exists. As the two-hemisphere model serves as a bridge between problem domain and software design phase, business model is understandable to both - business people and developers.

The inclusion of concept model in the approach is motivated by principles of object-oriented paradigm and general context of data analysis. Usually, at the beginning of software development data dictionary is created or there is any other agreement about terminology used in software development and documentation. [24] describes conceptual modelling as basis of software development, without which good design cannot be performed. Conceptual models are high-level software description, which contains concepts. Any kind of things, events and living beings that are important to given problem domain can be considered as concepts. Concepts are described with attributes, but methods shows actions specific to these concepts. Peter Chen's [25] Entity-Relationship (ER) diagram as mentioned by [26] was used in database design, but later it was also used in software system design as conceptual model. [27] indicates, that nowadays it is topical to use ontology not only in artificial intelligence (robot, agent) systems, but also in to create unified terminology, so that all stakeholders could communicate. [28] shows that ontology represents classes of objects, class relations, attributes and axioms. It provides a basis for choosing to use model that represents problem domain concepts as the other model. Therefore the other diagram of two-hemisphere model is conceptual model, consisting of concepts and its attributes, where model notation is similar to the same in ER diagram.

4 2HMD Approach for Solving the Task of Composition

The strategy of 2HMD approach supports gradual model transformation from problem domain models into program components, where problem domain models reflect two fundamental things: system functioning (processes) and structure (concepts and their relations). The two hemisphere model has been marked as input with mapping rules, the class diagram and transformation trace has been received on output (see Figure 2). Transformation trace shows the plan how an element of the two hemisphere model is transformed into the corresponding element of the class diagram, and which parts of the mapping are used for transformation of every part of the two hemisphere model [29].

The model decomposition into small components and composition of them as a whole system is a new research topic for 2HMD approach, originally introduced in [30]. The work is ongoing development and evolution. So, there is still no mature foundation to date for this. Our goal through the research on the composition of CPS is to study existing models of composition approaches by analyzing and identifying 1) what are the elements involved in the composition process, and 2) how the model composition is made in these approaches. The ultimate goal is to arrive at an understanding of what is done for model composition in these approaches.

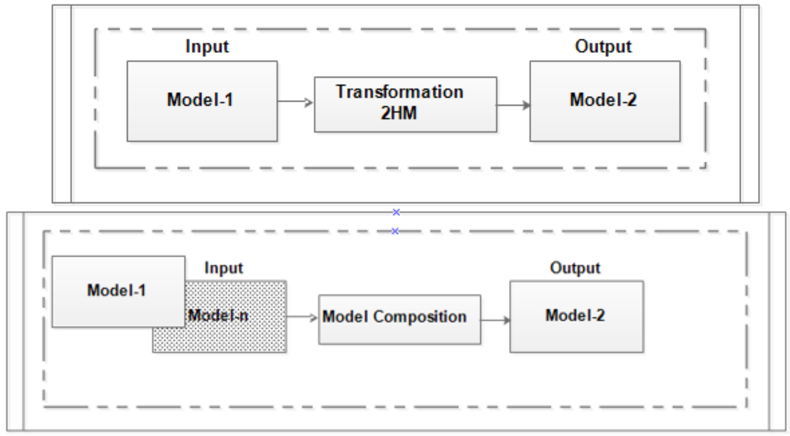


Fig. 2. Strategy for Composition

"Model composition is an operation that combines two or more models into a single one." [31] "Model composition in its simplest form refers to the mechanism of combining two models into a new one." [32]

According to this, it can be said that the composition model is a process that takes two or more input models, integrates them through an operation and composition to produce a composite output model.

However, this scheme is very abstract. No assumptions about the input models, output, or on the compositing operation is expressed. In practice, each approach must specify these assumptions for its work context. These also include the differences to classify approaches.

- Mechanism of composition: melting, replacing the union, weaving etc. Element composition: what are the additional elements involved in the composition. There are two classification axes: the type and formality of these.
- Language of composition: The composition of elements need formalisms to express them. These formalisms are very diverse because each approach has its own elements of composition. They can be a weaving language, a metamodel of composition rules, a UML profile for model composition, etc [32]. Despite their diversity, they can usually assess a compositional formalism on two points: the composition that provides abstractions and scalability.

The idea of decomposition methodology for classification tasks is to break down a complex classification task into several simpler and more manageable subtasks that are solvable by using existing induction methods, then joining their solutions together in order to solve the original problem. Decomposition methodology can be considered as an effective strategy for changing the representation of a classification problem. Indeed, [33], [34] considers decomposition as the most useful form of transformation of data sets".

This decomposition can be applied using a Multi-model approaches: when tackling the complexity of large software systems, separation of concerns is essential for keeping the development process, the produced models and the code manageable. The separation of concerns can be done in different ways, but the objectives are always the same: being able to identify relatively independent "parts" [35], [36].

To synthesize, we can define the composition as a model management operation, which generates a single model by the combination of the contents of at least two models. The composition has an impact on three levels:

1. Syntactic level: Expression model compound from input models.
2. Semantics level: Assigning a semantic model compound, depending on the semantics of the associated source models.
3. Methodical level: Using the model compound, derived from the composition process in a software development process.

Therefore, the composition process cannot be considered as an atomic operation. Before triggering the composition process itself, it is necessary to identify the links between the elements composing; hence the emergence of the pre-match phase followed by a composition operation that aims at the creation of the model "global" by combining elements using input patterns of relationships defined in the matching pattern. So considering all these all these criteria it's clear that making a survey on composition techniques and identify their gaps seems an interesting path to build a new composition models operations based on two hemisphere model approach. In another side we suggest using this taxonomy to create a novel composer framework to resolve composition conficts for a given problem. So now we are also studying the made to take into account the semantic properties of models. If we take the example of two operations in two models that appear with the same signature (name, type, parameters), so to remedy this problem, we must either include a step of reconciliation between the separate designs or strengthen semantics associated with the input metamodel, so that we can implement finer comparison strategies that address the behaviors described by the methods.

5 Conclusion

The evolutionary nature of CPS aims at building cross-domain intelligence, in heterogeneous and dynamic contexts [37] [38]. For this reason, CPS decomposition should focus on the interactions between the control logic and the physical systems, contemplating the possibility of limited information, e.g., stability, safety, performance, timeliness, etc.

CPS decomposition can be performed according to different criteria, from the CPS itself which is a schema of CPS as systems of systems [39], to a hierarchy of components at the architectural level [40]. The common feature among the different decomposition approaches is how they encapsulate the cyber and physical aspects through an infrastructure. The latter should allow the separation of

CPS concerns and, also provide support for the orchestration of a larger system architecture.

In turn to manage the process of CPS decomposition the strategy of CPS components' composition became the important task for CPS modelling. The model composition is the central concept also in model driven architecture for maximizing return on investment, dealing with complexity and maintainability. This paper discuss abilities on adopting a new methodology presented in the form of a conceptual prototype to automatically compose models defined in terms of class diagrams in order to build a global view of the system under construction. We have presented the progress process on model composition and based on the two hemisphere model driven approach introduced in [1].

The idea is focusing on model composition paradigm as a crucial activity. The composition of CPS is applied based on two-hemisphere model driven approach, which is an approach that aims to automate the process of class diagram development from correct and precise two-hemisphere model and enables knowledge representation in a form understandable for both business users and system analyst. As far as the two-hemisphere model-driven approach allow to share responsibilities among object classes and to define the relationships between them, we can consider that for CPS we can define:

1. the general schema of their components (the same as classes for object-oriented system),
2. how to share responcibilities between them, i.e., to define which processes will be performed by which components,
3. structural relationships among CPS components (as well as dependencies) within the task of their implementation.

The central hypothesis of the two-hemipshere model-driven approach is to apply many transformations for composition of the complex system, where the source model is defined in terms of a business process model, associated with a concept model, and the target model is defined in terms of class diagram. In this paper authors tried to investigate the possibility to continue from the point which the two hemisphere model fiish by using model composition and to compose produced class diagrams in order to automate the whole process and help in mitigating ever-growing complexity of modern software system. When the models are small enough and developed by a single or a couple of designers, they can be composed manually. However, in the case of CPS, the models are too large to be composed manually and it's necessary to develop an automatic composition method to ensure that all the elements in the model are handled. In this paper we have only taken into account the conceptual part of our methodology, thereby the authors try at the moment to investigate the possibility to implement the proposed technique as an open source tool using ATL Language. The idea of conceptual composition prototype described in this paper currently can handle not only homogeneous models, those that share the same meta-model, it would be interesting to extend this approach to handle heterogeneous input models as well. Also, we are currently dealing only with one-to-one match relationships between input model elements. We plan to investigate the extension of our approach for handling many-to-many match relationships. We also plan to extend our prototype to provide a better support for the interactive weaving process, and captures positive and negative result of previous interactions.

As a future work to what is presented in this paper we are currently investigating a finer-grained redefinition of every module separately, the first one will be a repository dedicated to the resolution of potential composition conflicts. This allows focusing on any type of conflicts that requires special treatment, thereby it will facilitate the generic implementation of the other modules. Another line of future investigations concerns the model comprehension aspects of our model composition technique. The benefits for model comprehension address in particular the reverse process of building model hierarchies.

Acknowledgments. The research is performed as a result of cooperation within the support of European COST Action IC1404 for Multi-Paradigm Modelling for Cyber-Physical Systems (MPM4CPS).

References

1. Nikiforova, O., Kirikova, M.: Two-Hemisphere Driven Approach: Engineering based Software Development. In: Advanced Information Systems Engineering, Proceedings of the 16th International Conference CAiSE 2004, Riga, Latvia, June 2004, A. Persson and J. Stir-na (Eds.), Springer Verlag Berlin Heidelberg, 2004, ISBN 3-540-22151-4, pp. 219-233 (2004)
2. Nikiforova, O., Kirikova, M., Wojtkowsk,i W.: Role of models in knowledge transfer during OO software development. In Proceedings of the 15th European-Japanese Conference on Information Modelling and Knowledge Bases: Tallinn, Estonia, May 16-20, 2005, Y.Kiyoki, H.Kangassalo, H.Jaakkola and J.Henno (Eds.), ISBN 9985-59-530-0, pp. 305-320 (2005)
3. "US National Science Foundation, Cyber-Physical Systems (CPS)", https://www.nsf.gov/funding/pgm_summ.jsp?pims_id=503286
4. Khaitan, et al.: Design Techniques and Applications of cyber-physical Systems: A Survey. In: IEEE Systems Journal (2014)
5. Gribaudo, M., Iacono, M.: An introduction to multiformalism modelling. In: Theory and applications of multi-formalism modelling, IGI-Global, ISBN13: 9781466646599, ISBN10: 1466646594, DOI: 10.4018/978-1-4666-4659-9.ch001
6. Vangheluwe, H., De Lara, J., Mosterman, P. J.: An introduction to multi-paradigm modelling and simulation. In: Proceedings of the AIS'2002 conference (AI, Simulation and Planning in High Autonomy Systems), Lisboa, Portugal, pp. 9-20 (2002)
7. Barbierato, E., Gribaudo, M., Iacono, M.: Exploiting multiformalism models for testing and performance evaluation in SIMTHESys. In: Proceedings of 5th International ICST Conference on Performance Evaluation Methodologies and Tools - VALUETOOLS 2011, Cachan (France), May 16-20 (2011), ICST
8. Barbierato, E., Gribaudo, M., Iacono, M.: modelling Hybrid Systems in SIMTHESys. In: Electronic Notes on Theoretical Computer Science, vol. 327, pp. 5-25 (2016), Elsevier

9. Barbierato, E., Dei Rossi, G., Gribaudo, M., Iacono, M., Marin, A.: Exploiting product forms solution techniques in multiformalism modelling. In: Electronic Notes in Theoretical Computer Science, vol. 296, pp. 61-77 (2013), Elsevier
10. Hancu, O., Maties, V., Balan, R., Stan, S.: Mechatronic approach for design and control of a hydraulic 3-dof parallel robot". In: The 18th International DAAAM Symposi-um, "Intelligent Manufacturing & Automation: Focus on Creativity, Responsibility and Ethics of Engineers" (2007)
11. Lee, E.A., Seshia, S.A.: Introduction to Embedded Systems - A Cyber-Physical Systems Approach. LeeSeshia.org (2011)
12. Suh, S.C., Carbone, J.N., Eroglu, A.E.: Applied Cyber-Physical Systems. Springer (2014)
13. Rad, C.-R., Hancu, O., Takacs, I.-A., Olteanu, G.: Smart Monitoring of Potato Crop: A Cyber-Physical System Architecture Model in the Field of Precision Agriculture. In: Conference Agriculture for Life, Life for Agriculture. 6: 73-79 (2015)
14. Bure, T., Gerostathopoulos, I., Hnetynka, P., Keznikl, J., Kit M., Plail, F.: DEECo - an Ensemble-Based Component System. In: Proceedings of CBSE 2013, Vancouver, Canada (2013)
15. Keznikl J., Bure T., Plail F., Kit M.: Towards Dependable Emergent Ensembles of Components: The DEECo Component Model. In: Proceedings of WICSA/ECSA 2012, Helsinki, Finland, pp. 249-252, IEEE CS, ISBN 978-0-7695-4827-2, DOI 10.1109/WICSA-ECSA.212.39, (2012)
16. Hennicker, R., Klarl, A.: Foundations for Ensemble modelling - The Helena Approach. In: Specification, Algebra, and Software, Volume 8373 of the series Lecture Notes in Computer Science, pp. 359-381 (2014)
17. Al Ali, R., Bure, T., Gerostathopoulos, I., Keznikl, J., Plail, F.: Architecture Adaptation Based on Belief Inaccuracy Estimation. In: Proceedings of the 11th Working IEEE/IFIP Conference on Software Architecture (WICSA 2014), Sydney, Australia. IEEE, pp. 87-90 (2014)
18. Bure T., Hnetynka P., Kofron J., Al Ali R., koda D.: Statistical Approach to Architecture Modes in Smart cyber-physical Systems, Proceedings of WICSA 2016, Venice, Italy, IEEE, pp. 168-177, doi: 10.1109/WICSA.2016.33 (2016)
19. Ould, M. A.: Business Process Management: A Rigorous Approach, British Computer Society (2005)
20. Badica, A., Badica, C., Leon, F., Buligiu, I.: modelling and Enactment of Business Agents Using Jason. In: Proceedings of the 9th Hellenic Conference on Artificial Intelligence, SETN 2016, Thessaloniki, Greece (2016) doi: 10.1145/2903220.2903253.
21. Leon, F., Badica, C.: A Comparison Between Jason and F# Program-ming Languages for the Enactment of Business Agents, Proceedings of the Interna-tional Symposium on INnovations in Intelligent SysTems and Applications, INISTA 2016, Sinaia, Romania (2016) doi: 10.1109/INISTA.2016.7571841
22. Polak, P.: BPMN Impact on Process Modelling. Proc. of the 2nd International Busi-ness and Systems Conference BSC, Riga, Latvia, November 5, pp. 26-35 (2013)
23. Harmon, P, Wolf, C.: The State of Business Process Management, BPTrends (2014) http://www.bptrends.com/
24. Johnason, J., Henderson, A.: Conceptual Models. Core to Good Design, 1st edition. Morgan & Claypool Publishers, 2011. 110 p.
25. Chen, P.P.: The Entity-Relationship Model: Toward a Unified View of Data. A CM Transactions on Database Systems, vol.1, pp.9-36 (1976)
26. Hesse, W.: Ontologies in the Software Engineering process. Proc. of the 12th Interna-tional Workshop on Exploring Modelling Methods for Systems Analysis and Design (EMMSAD-2007), Trondheim, Norway, 11-15 June, pp.1-13 (2007)

27. Graudina, V., Grundspenkis, J.: Algorithm of Concept Map Transformation to On-tology for Usage in Intelligent Knowledge Assessment System. Proc. of the 12th Interna-tional Conference on Computer Systems and Technologies, Vienna, Austria, June 16-17 (2011)
28. Mdche, A., Schnurr, H.P., Staab, S., Studer, R.: Representation-Language-Neutral Modelling of Ontologies. J. Ebert, U. Frank (Hrsg.): Modelle und Modellierungssprachen in Informatik und Wirtschaftsinformatik. Proc. "Modellierung 2000". Koblenz: Flbach-Verlag, pp.143-150 (2000)
29. Nikiforova O., Two Hemisphere Model Driven Approach for Generation of UML Class Diagram in the Context of MDA // e-Informatica Software Engineering Journal - Vol-ume 3, Issue 1, Huzar Z., Madeyski L. (eds.), Wrocaw University of Technology, Institute of Applied Informatics, Wrocaw University of Technology, Wrocaw, Poland, Copyright by Oficyna Wydawnicza Politechniki Wrocawskiej, Wrocaw, Poland, pp. 59-72, 2009, available at http://www.e-informatyka.pl/wiki/e-Informatica_-_Volume_3
30. El Marzouki, N., Nikiforova, O., Lakhrissi, Y., El Mohajir, M., Enhancing Conflict Resolution Mechanism for Automatic Model Composition. In: Scientific Journal of Riga Technical University: Applied Computer Systems, Grundspenkis J. et al. (Eds), Vol.19, pp. 44-52 (2016)
31. Cavallaro, L., Di Nitto, E., Furia, C.A., Pradella. M.: A tile-based approach for self-assembling service compositions. In: Engineering of Complex Computer Systems (ICECCS), 15th IEEE International Conference on, pages 43-52, IEEE (2010).
32. Dube, M. R., Dixit, S.K.: modelling Theories and Model Transformation. In: International Journal of Computer Applications (0975-8887), Volume 38 - No. 7, January 2012, pp. 11-19 (2012)
33. Acher, M., Collet, P., Lahire, P., France, R.: Comparing approaches to implement fea-ture model composition. In: Modelling Foundations and Applications, pages 3-19 (2010).
34. Clasen, C., Jouault, F., Cabot. J.: Virtualemf: a model virtualization tool. Advances in Conceptual modelling. In: Recent Developments and New Directions, pages 332-335 (2011).
35. He, D. W., Strege, B., Tolle, H., Kusiak, A.: Decomposition in Automatic Generation of Petri Nets for Manufacturing System Control and Scheduling. In: International Journal of Production Research, 38(6): 1437-1457 (2000)
36. Kusiak, A.: Decomposition in Data Mining: An Industrial Case Study. In: IEEE Transactions on Electronics Packaging Manufacturing, Vol. 23, No. 4, pp. 345-353 (2000)
37. Khaitan, S. K., McCalley, J. D.: Design techniques and applications of cyber-physical systems: A survey, IEEE Systems Journal 9 (2) (2015) pp. 350-365
38. Wu, F.-J., Kao, Y.-F., Tseng, Y.-C.: From wireless sensor networks towards cyber-physical systems. In: Pervasive and Mobile Computing 7 (4) (2011) pp. 397-413
39. Nazari, S., Sonntag, C., Engell, S.: A modelica-based modelling and simulation framework for large-scale cyber-physical systems of systems, IFACPapersOnLine 48 (1) (2015) pp. 920-921
40. Bhave, A., Krogh, B. H., Garlan, D., Schmerl, B.: View consistency in architectures for cyber-physical systems. In: Cyber-Physical Systems (ICCPS), 2011 IEEE/ACM International Conference on, IEEE, 2011, pp. 151-160 (2011)

Consistency and Uncertainty in the Development of Cyber-Physical Systems

Antonio Cicchetti

School of Innovation, Design and Engineering
Mälardalen University, SE-721 23, Västerås, Sweden
antonio.cicchetti@mdh.se

Abstract. Cyber-physical systems (CPS) expose software development to the intricacy of dealing with the physical world. Due the ever increasing complexity of CPS and the ubiquity of software, it is no more feasible to adopt separate development processes of the *cyber* and *physical* domains. In fact, those two get intertwined as one acting on the other in an undistinguishable manner. The current development practices typically enforce a waterfall process to keep the consistency between design decisions from different domains, which however makes the lead time very long. More importantly, they impose to make premature design decisions that can cause costly reworks.
In this paper we argue that a better process should support tolerating inconsistencies and uncertainty in the design and development of CPS. The final aim is to allow postponing important design decisions to only when the system is mature enough, i.e. when there is information at hand to take evidence-based decisions, while still keeping a desired level of development *convergence*. In this respect, we outline several research directions and open challenges towards the support of inconsistency tolerance and uncertainty in the development of CPS.

1 Consistency and Uncertainty in the development of CPS

Cyber-Physical Systems (CPS) are truly complex designed systems, since they result from the combination of software, physical, and networking parts. Developing these multidisciplinary systems-of-systems can be several orders of magnitude more complex than the common embedded systems. At present, no unified systematic method or theory exists to tackle their development endeavour [1].

In general, design and development processes of CPS tend to proceed in *silos* and cross-disciplinary interactions are reduced as much as possible due to their complexity [2–4]. In particular, a widely adopted development strategy (not only for CPS) is to partition the system in smaller parts, and each sub-portion of the system is developed in isolation. When a certain development stage is considered as completed, the system specification at hand is frozen to allow further, lower level, development steps. Whenever modifications impacting the current specification are required, they are left pending in the form of change requests for the next development process iteration. In this way, it is possible to enforce a certain level of system-wide consistency. However, development processes organized in silos tend to be very inefficient, as the lead time of a project can span over years. Moreover, especially in the case of CPS, cross-disciplinary inconsistencies might be introduced and stay hidden, at worst, until the actual product is fully assembled and tested [5].

Multi-Paradigm Modelling (MPM) [6] proposes to alleviate the intricacy of CPS development by embracing the *model everything* principle: each aspect of the system is specified by means of models at the appropriate level of abstraction, while information exchanges are supported through model transformations. In this way, models can play the role of a *lingua franca*, able to support cross-disciplinary communication and collaboration. Despite MPM potentials, its adoption faces relevant theoretical and practical obstacles that require more mature techniques before the possible application in industrial setting. Notably, the accidental complexity to keep the system in a consistent state grows remarkably, due to both the amount of details to be considered and the sophistication of dependencies between the various aspects of a CPS.

In this paper, we focus on two issues that we consider as fundamental for enabling the adoption of MPM methods in the development of CPS: i) consistency management should be relaxed in order to alleviate the (accidental) complexity introduced due to pursuing full traceability; ii) design techniques should support uncertainty, since it is not rare that important design decisions have to be taken to let the process go forward, but the system is not mature enough for permitting the evaluation of all the possible alternatives at hand.

It is worth noting that the mentioned research directions are closely related: uncertainties have to be kept under control to ensure that the space of possible solutions is constrained by feasibility of the CPS. We define the solutions in such a set as convergent, an propose to exploit inconsistency tolerance to control the solution space. The remainder of this contribution illustrates ongoing work towards the mentioned issues together with related open challenges.

2 Outline: Research Agenda and Challenges

The complexity of modern systems is continuously growing and software plays a fundamental role in such systems. In this respect, companies have been forced to introduce more and more software development, even in those cases where software was not the main product of the company. It is enough to consider that a CPS like a modern car embeds near to one hundred processing units, controlling every functionality of the car, ranging from the propulsion and braking systems to the entertainment features [7].

Keeping the consistency among software, hardware, and network aspects of a CPS is a challenge, given the intrinsic semantic gaps between these aspects and their interactions [5]. As discussed in Section 1, by adhering to development approaches like MPM, this challenge can be alleviated by modelling each aspect at the proper abstraction level and then by creating automated consistency bridges through model transformations. In this respect, we believe that an applicable solution should support inconsistency and uncertainty, which are inherent factors in CPS development.

2.1 On the Tolerance of Inconsistencies

The development of CPS is necessarily distributed, thus entailing the need for keeping the resulting system specification consistent throughout the process. However, the lessons learned in practice suggest to relax consistency relations in order to allow a more effective development approach. In particular, in general the system might still be at an embryonic state to establish full consistency between the different aspects it is made up of. Moreover, it can happen that some of the inconsistencies naturally disappear as the result of the development.

We propose to relax consistency management by exploiting the notion of inconsistency tolerance, that is the system specification has to be consistent when checked for it, but can be inconsistent otherwise [8, 9]. By using this approach it is possible to establish when a system must be consistent and how such a state needs to be tested. It is worth noting that the terms by which consistency is checked implicitly define how important is to keep consistency between certain aspects of the system specification.

We have produced preliminary results towards the support of inconsistency tolerance for the development of CPS: in [10] we have discussed the concept of semantic inconsistency and its tolerance. By going into some more details, checking the consistency at semantic level is needed due to the heterogeneous/cross-disciplinary characteristic of CPS sub-systems. Based on this, we introduced inconsistency tolerance relationships as based on the notion of distance between traces of the properties taken into account. In this respect, inconsistencies are tolerated until their distance is limited to a certain threshold, over which consistency needs to be restored in order to keep system development convergent.

Inconsistency tolerance does not only save the efforts of continuously keeping the system in a consistent state, but it also save all the resources required to check the consistency itself: in fact, property measurements might be time-consuming and the (sub-)system development has to be necessarily stopped in the meanwhile.

2.2 On the Support of Uncertainty

In general, it might happen that some design decisions cannot be taken due to several factors, notably the early stage of development, unknown details, or more simpler because two or more alternatives appear as equally suitable in a specific system development status. In such cases it is desirable to have modelling features to represent the uncertainties rather than being forced to take a premature decision.

Uncertainty can be considered as an aspect orthogonal to inconsistencies. In fact, inconsistencies might have multiple valid resolutions, which however have different impacts on other aspects of the system being developed [10]. This becomes evident for CPS, given the semantic gaps between the various domains involved in the realization.

As initial contribution, in [11] we proposed an approach to deal with uncertainty in the development of automotive systems. In particular, starting from a high level architecture of the system, multiple valid lower level configurations can be derived. These configurations however have different impacts on timing performances of the system. In order to avoid imposing a premature decision, often only based on development history experience, we introduced a modelling support for keeping equally good configurations to choose from at later stages of the development.

2.3 Further Research and Challenges

What discussed so far can be considered as initial steps towards the support of inconsistency tolerance and uncertainty in the development of CPS. As expected, a multitude of problems remain to be solved, as well as open research questions, a selection of which is described in the remaining of the section.

The misalignment of development methodologies As a consequence of the growth in complexity of delivered products, companies try to adopt more effective development methodologies. However, on the one hand there might not exist mature enough techniques for tackling a certain industrial problem. For instance, the availability of effective tools is still a major hinder for model-driven engineering adoption in industry [12]. On the other hand, for a number of practical reasons such adoptions are step-wise, that is the various departments progressively make a transition towards a newer approach depending on their specific needs, resources, constraints, and so forth.

Different development approaches and tools in general entail different levels of abstraction, formalisms, and so forth. This misalignment is a fundamental challenge in CPS development, since it directly affects the effectiveness of consistency management: the wider is the abstraction and semantic gap between different development approaches and the harder it becomes to create an effective consistency support. Moreover, if those gaps have to be manually closed, consistency is jeopardized due to the tediousness and error-proneness of these tasks [13].

The growth of accidental complexity A well known issue intrinsic of any development methodology is the added complexity due to the use of the methodology itself, know as accidental complexity [14]. When systems become complex, as in the case of CPS, the accidental complexity tends to grow remarkably, especially when trying to achieve reliable analysis and simulations results.

Separation of concerns is usually recognized as an effective way to reduce accidental complexity, since it narrows down the problem space to a specific point-of-view of the system. However, this solution often boils down to moving accidental complexity from the design of the various aspects of the system to the consistency mechanisms. As a matter of fact, in general each pair of views requires a dedicated, semantics-aware, consistency management. Therefore, the support becomes quickly intractable with the growth of the number of views.

A countermeasure to reduce accidental complexity could be the introduction of a (set of) language(s) acting as common denominator(s) and hence allowing automated translations and information transfers between different system points-of-view. However, the characteristics of such languages are still an open research question.

The solution space Both inconsistency tolerance and uncertainty rely on leaving open multiple solution alternatives. For CPS such opportunity has to be kept carefully under control, since the size of the solution space might make the proposed techniques practically unusable. In fact, given the number of variables and possible trade-offs, the number of valid alternatives can become unmanageable, especially when decisions cannot be automated, with the risk of obtaining non convergent (infeasible) design options.

Another more practical aspect to be considered is that the development of portions of CPS might be outsourced. In such cases uncertainty is not tolerable, since typically the development is bound to a strict contract.

The issues mentioned so far could be alleviated by introducing a more suitable development process. Such a process should be CPS aware in the sense that it should allow to establish priorities among system aspects and hence allow a better management of both inconsistencies and uncertainty. In particular, the design space entailed by uncertainties can be constrained by inconsistency tolerance boundaries, where tighter tolerance boundaries mean reduced uncertainty allowance. Also in this case, the subject is an open research question.

3 Conclusions

In this paper we discussed some features that we believe it is necessary to support for improving the development of CPS. In particular, we argue that inconsistency tolerance and uncertainty modelling can make the development process smoother and alleviate some issues in the development of complex, cross-disciplinary systems.

Despite the growing maturity of modelling methodologies like MPM, there exist still several open research questions to be tackled. These challenges pertain to both scientific concerns as the tractability of complex problems, as well as to more practical aspects as the technological transfer from research to industry [1].

4 Acknowledgements

The author would like to thank Alessio Bucaioni, Federico Ciccozzi, and Alfonso Pierantonio for the interesting discussions on uncertainty support. Moreover, he would like to thank Dominique Blouin, István Dávid, Eugene Syriani, and Ken Vanherpen for the insightful idea exchanges on the inconsistency tolerance topic during the 2016 CAMPaM workshop.

References

1. Vangheluwe, H., Amaral, V.: Memorandum of Understanding for the implementation of a European Concerted Research Action designated as COST Action IC1404: Multi-Paradigm Modelling for Cyber-Physical Systems (MPM4CPS). http://www.cost.eu/COST_Actions/ict/IC1404 (2014)
2. Pradhan, S.M., Dubey, A., Gokhale, A., Lehofer, M.: CHARIOT: A Domain Specific Language for Extensible Cyber-physical Systems. In: Proceedings of the Workshop on Domain-Specific Modeling. DSM 2015, New York, NY, USA, ACM (2015) 9–16
3. Schätz, B., Törngren, M., Bensalem, S., Cengarle, M.V., Pfeifer, H., McDermid, J., Passerone, R., Sangiovanni-Vincentelli, A.: D6.1+2 - Integrated CPS Research Agenda and Recommendations for Action. http://cyphers.eu/sites/default/files/d6.1+2-report.pdf (2015)
4. Ciccozzi, F., Crnkovic, I., Di Ruscio, D., Malavolta, I., Pelliccione, P., Spalazzese, R.: Model-driven engineering: a facilitator for engineering mission-critical IoT systems. IEEE Software (2016) 1–14

5. Persson, M., Törngren, M., Qamar, A., Westman, J., Biehl, M., Tripakis, S., Vangheluwe, H., Denil, J.: A characterization of integrated multi-view modeling in the context of embedded and cyber-physical systems. In: Proceedings of the Eleventh ACM International Conference on Embedded Software. EMSOFT '13, Piscataway, NJ, USA, IEEE Press (2013) 10:1–10:10
6. Mosterman, P.J., Vangheluwe, H.: Computer automated multi-paradigm modeling: An introduction. SIMULATION **80** (2004) 433–450
7. Mossinger, J.: Software in automotive systems. IEEE Software **27** (2010) 92–94
8. Balzer, R.: Tolerating inconsistency. In: 13th International Conference on Software Engineering. (1991) 158–165
9. Finkelstein, A.: A foolish consistency: Technical challenges in consistency management. In Ibrahim, M., Küng, J., Revell, N., eds.: Database and Expert Systems Applications. Volume 1873 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2000) 1–5
10. Dávid, I., Syriani, E., Verbrugge, C., Buchs, D., Blouin, D., Cicchetti, A., Vanherpen, K.: Towards inconsistency tolerance by quantification of semantic inconsistencies. In: Procs. of the 1st Int. Workshop on Collaborative Modelling in MDE (COMMitMDE 2016) at MoDELS 2016, Saint Malo (France). CEUR-WS, CEUR Workshop Proceedings (2016)
11. Bucaioni, A., Cicchetti, A., Ciccozzi, F., Mubeen, S., Sjödin, M., Pierantonio, A.: Handling uncertainty in automatically generated implementation models in the automotive domain. In: Procs. of the 42nd Euromicro Conference series on Software Engineering and Advanced Applications (SEAA 2016), Limassol (Cyprus), IEEE Computer Society (2016)
12. Whittle, J., Hutchinson, J., Rouncefield, M., Burden, H., Heldal, R.: Industrial adoption of model-driven engineering: Are the tools really the problem? In: Proceedings of the 16th International Conference on Model-Driven Engineering Languages and Systems - Volume 8107, New York, NY, USA, Springer-Verlag New York, Inc. (2013) 1–17
13. Selic, B., Gullekson, G., McGee, J., Engelberg, I.: Room: an object-oriented methodology for developing real-time systems. In: Computer-Aided Software Engineering, 1992. Proceedings., Fifth International Workshop on, IEEE (1992) 230–240
14. Brooks, Jr., F.P.: No silver bullet. essence and accidents of software engineering. Computer **20** (1987) 10–19

Multi-paradigm aspects of component ensembles

http://d3s.mff.cuni.cz

Tomas Bures

bures@d3s.mff.cuni.cz

CHARLES UNIVERSITY IN PRAGUE

Faculty of Mathematics and Physics

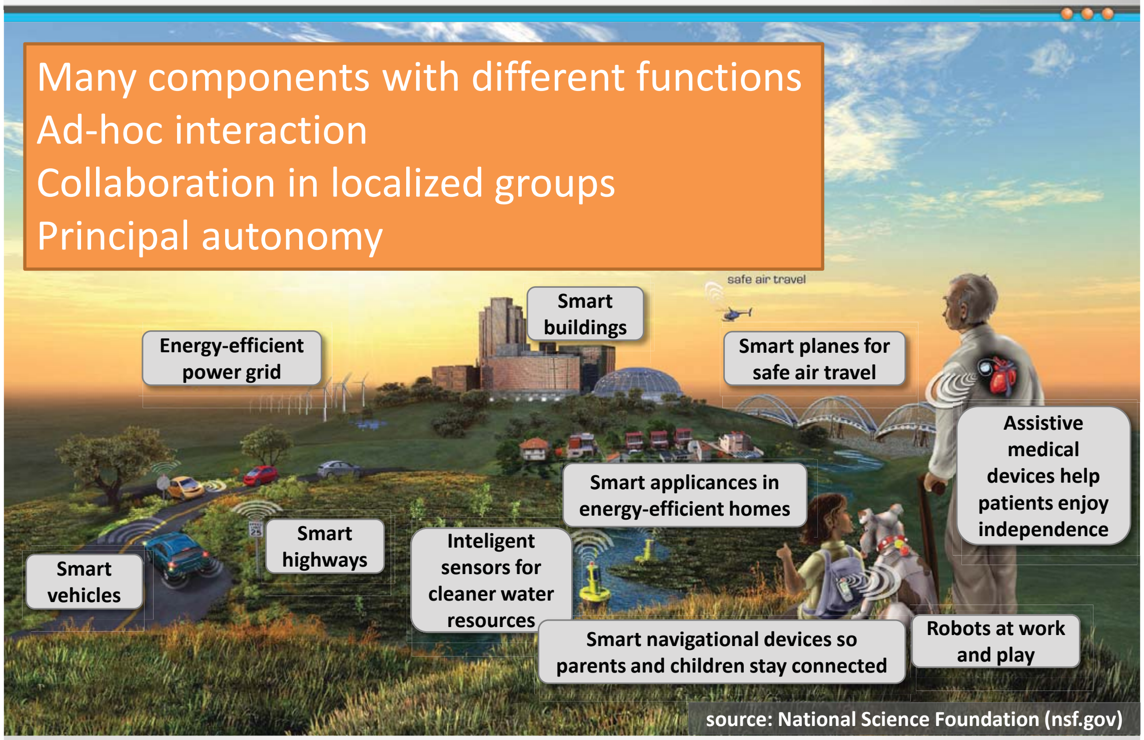
Department of Distributed and Dependable Systems

Context: Software Architectures

- Software composed of components
- Large-scale distribution, mobility
- Runtime evolution of the architecture
  - Software adapts to environment (environmental uncertainty)
  - Software adapts based on health of the system (internal uncertainty)
  - Connections change due to mobility

System of Interacting Components

Many components with different functions
Ad-hoc interaction
Collaboration in localized groups
Principal autonomy



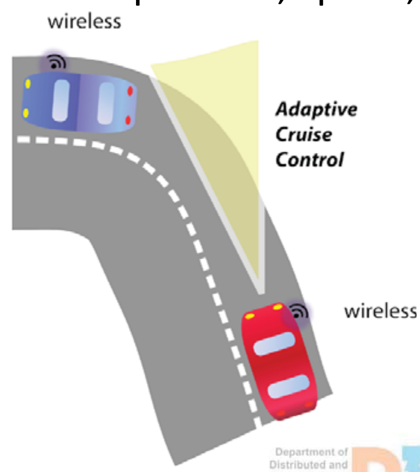
source: National Science Foundation (nsf.gov)

Our Approach

- Dynamic architecture models (ADLs)
  - Components
  - Ensembles – dynamic collaboration groups
- Architecture is the central hub for MPM; in addition to structure, it:
  - Says how the systems evolves and how its different component collaborate on achieving system's goals
  - Provides domain knowledge that allow optimizing distributed (e.g. MANET) communication
  - Specifies physical models of data being observed
    - Allows for filtering and prediction
    - Reconfiguration of architecture if data become faulty

Example

- Cooperative Adaptive Cruise Control-like system
  - A car follows another, keeps in a safe distance behind it
  - The car in front sends over wireless its position, speed, acceleration
  - The car in the back does dead-reckoning to know whether it should adapt its behavior (i.e switch architecture mode)




```
component Vehicle
  // leader's role
  knowledge:  position, velocity, ...
  state-space-models:  [leader.position, leader.velocity]:
    maxDecTable = {0 -> -6, 35 -> -5, 51 -> -3}
    maxAccTable = {0 ->  4, 35 ->3, 51 -> 0}
process measurePosition(out position):
    scheduling: periodic( 100ms  )
  process driveUsingCACC(…):
    mode-trigger: possible-min (distance(position, leaderPosition)) <= THRESHOLD
  process computeAccelerationACC(in distance, in velocityDifference, out targetAcc):
    mode-trigger: possible-min(distance(position, leaderPosition)) > THRESHOLD
  …
```

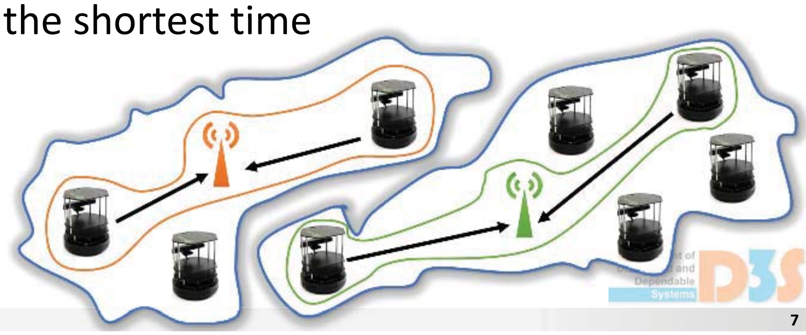
Linear state space model of car's movement
Represented here as tables for lookup of max. deceleration/acceleration based on current velocity.
The tables consist of tuples (velocity (in m/s) -> acceleration (in m/s2))

```
ensemble UpdateLeaderPositionAndVelocity
  membership
    roles
      leader: Vehicle
      follower: Vehicle
    condition
      distance(coordinator.position, member.position) ≤ 2 * DESIRED_DISTANCE
  knowledge exchange
    coordinator.leaderPosition ← closest(coordinator.position, members).position
    coordinator.leaderVelocity ← closest(coordinator.position, members).velocity
  scheduling periodic( 200ms )
```

Dynamic cooperation group.
Comes to existence when a car in the back reches the car in front.

Example

- Robot exploration on a set of islands
  - Dynamically appearing beacons on a set of islands
  - A number of robots deployed on the islands
  - Robots exchange information about beacons they have discovered
  - Each beacon has to be reached and "handled" by a pair of robots in the shortest time




```
component Robot
 knowledge
  position       // robot's position
  beaconPosition  // targeted beacon position
  islandID       // island, on which the robot is located
  beaconPositions // positions of known beacons

ensemble BeaconInformationExchange
 id islandID
 membership
  roles
   source: Robot
   target: Robot
  condition
   source != target
 knowledge exchange
   target.beaconPositions = target.beaconPositions.unionWith(source.beaconPositions)
 communication constraints
  boundary
   relay: RobotRelay, replica: Robot
   relay.islandID == replica.islandID
```

Restricts information exchange to an island only.
Reflects the domain knowledge that a robot can't get from one island to another.

```
ensemble ForSingleBeacon
 id beaconPosition    // targeted beacon position
 membership
  roles
   robotsAssignedForBeacon[2]: Robot
  condition
   robotsAssignedForBeacon[0].islandID == robotsAssignedForBeacon[1].islandID ==
   islandIDOf(beaconPosition)
  fitness
   max(distance(robotsAssignedForBeacon[0].position, beaconPosition),
       distance(robotsAssignedForBeacon[1].position, beaconPosition))
 knowledge exchange
   robotsAssignedForBeacon[0].beaconPosition = beaconPosition
   robotsAssignedForBeacon[1].beaconPosition = beaconPosition
 communication constraints
  boundary
   relay: RobotRelay, replica: Robot
   relay.islandID == replica.islandID
  optimization
   smallestRadius > 10m
   max staleness beaconPostion 30s
```

Dynamic cooperation group.
Comes to existence when a beacon appears.
Selects the closest robots.

Provides bounds to communication optimization.
Expresses the domain knowledge that (a) in order for a system to work, robot pairs have to be looked up in at least 10 m radius around the beacon (reflects the expected density of robots; (b) robots move so fast that any information about robot's position older than 30s has no relevancy for the system.

9

Some References

- Masrur A., Kit M., Matěna V., Bureš T., Hardt W.: Component-Based Design of Cyber-Physical Applications with Safety-Critical Requirements, Accepted for publication in Microprocessors and Microsystems, March 2016
- Gerostathopoulos I., Škoda D., Plášil F., Bureš T., Knauss A.: Architectural Homeostasis in Self-Adaptive Software-Intensive Cyber-Physical Systems, In Proceedings of ECSA 2016, Istanbul, Turkey, LNCS 9839, Springer, to appear, September 2016
- Bureš T., Hnětynka P., Kofroň J., Al Ali R., Škoda D.: Statistical Approach to Architecture Modes in Smart Cyber Physical Systems, Proceedings of WICSA 2016, Venice, Italy, IEEE, pp. 168-177, doi: 10.1109/WICSA.2016.33, April 2016
- Matěna V., Bureš T., Gerostathopoulos I., Hnětynka P.: Model Problem and Testbed for Experiments with Adaptation in Smart Cyber-Physical Systems, In Proceedings of SEAMS 2016, Austin, USA, ACM, doi:10.1145/2897053.2897065, May 2016
- Kit M., Plášil F., Matěna V., Bureš T., Kovac O.: Employing Domain Knowledge for Optimizing Component Communication, Accepted for publication in Proceedings of the 18th International ACM Sigsoft Symposium on Component-Based Software Engineering, May 2015
- Bureš T., Krijt F., Plášil F., Hnětynka P., Jiráček Z.: Towards Intelligent Ensembles,
- In Proceedings of the 9th European Conference on Software Architecture Workshops (ECSAW 2015). Article No. 17. ACM., September 2015
- Al Ali R., Bureš T., Gerostathopoulos I., Keznikl J., Plášil F.: Architecture Adaptation Based on Belief Inaccuracy Estimation, Proceedings of the 11th Working IEEE/IFIP Conference on Software Architecture (WICSA 2014), Sydney, Australia. IEEE, pp. 87-90, April 2014

10

Statistical Approach to Architecture Modes in Smart Cyber Physical Systems

http://d3s.mff.cuni.cz

Tomas Bures, Petr Hnetynka, Jan Kofron, Rima Al-Ali, Dominik Skoda

CHARLES UNIVERSITY IN PRAGUE
Faculty of Mathematics and Physics

Department of Distributed and Dependable Systems
D3S

Context: Cyber-Physical Systems (CPS)

- Collaborating computational elements controlling physical entities
- Designed as a network of interacting elements with physical input and output
- Uncertainty

Context: Architectural modes

Traditional Architectural Using Modes

- Easy to Understand
- Simple and explicit specification
- Examples: AUTOSAR, AADL, ProCom, MyCCM-HI, SOFA-HI, ProCom

Missing

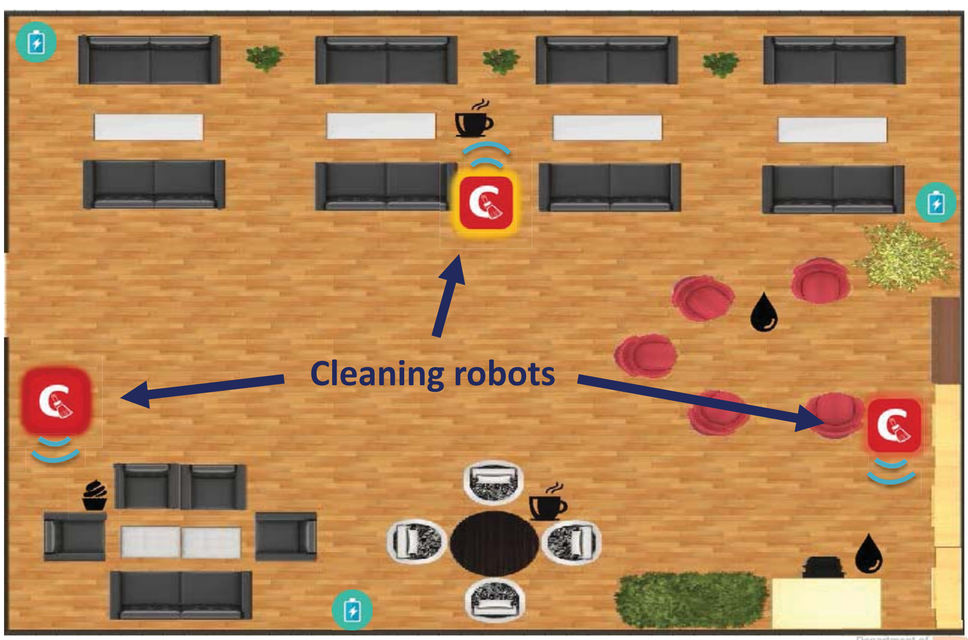
- Lack Expressivity of Uncertainty
- Depend on current state only and no trends or historical data are used

Our Contribution

- Statistical extension for mode switching to be more expressive toward uncertainty

3

Motivation example – Smart CPS

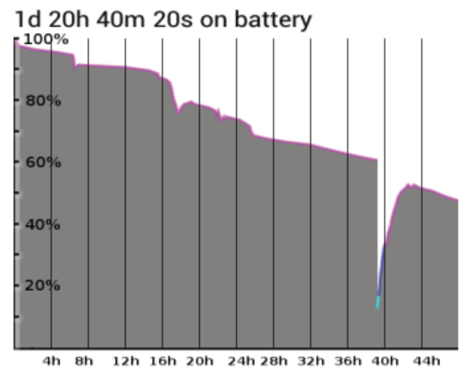


Smart Office – resting room with many cleaning robots.

4

Problem

- We have to deal and reason about uncertainty
  - "Switch to 'Go back to charger' mode
    … if energy level < 20%"
  - "Switch to 'Go and clean' mode
    …if the energy will suffice"



Sample battery energy level during continuous discharge

- Note:
  - We could filter the data, but still that does not help us in knowing how certain we are about a particular state

5

Problem

- Rather, we would like to say:
  - "Switch to 'Go back to charger' mode
    … if based on the recent data we can be 80% sure that energy level < 20%"
  - "Switch to 'Go and clean' mode
    …if based on the recent data we can be 95% sure that the energy will suffice"
  - "Switch to 'Roam around close to the charger' mode
    … if based on the recent data can't say anything with reasonable certainty"

6

Goal

- **Allow reasoning about uncertainty in mode switch guards.**

- Approach:
  - Logic for specifying mode guard based on statistical testing
  - Mapping to C++
  - Library for performing the tests

Department of Distributed and Dependable Systems D3S

7

Logic – Based on statistical testing

$\boldsymbol{F \leq_\gamma c, F_1 \leq_\gamma F_2}$ – if the null hypothesis $X \leq c$ or $X_1 \leq X_2$ respectively cannot be rejected at confidence level γ, where X, X_1, X_2 are random variables with distributions F, F_1, F_2 respectively .

$\boldsymbol{F <_\gamma c, F_1 <_\gamma F_2}$ – if the null hypothesis $X \geq c$ or $X_1 \geq X_2$ respectively can be rejected at confidence level γ, where X, X_1, X_2 are as above.

$\geq_\gamma$ **and** $>_\gamma$ are defined correspondingly

$=_\gamma$ is defined as $\leq_\gamma$ & $\geq_\gamma$

mean(A) – distribution used for comparing the sample mean of *A*

lra(A) – distribution used for comparing the value of intercept *α* in linear regression *α+βx* fitted to the time-series *A* via ordinary least squares (OLS)

lrb(A) – distribution used for comparing the slope *β*

lr(A,x) – distribution of *y=α+βx* as above

Department of Distributed and Dependable Systems D3S

8

Examples

- $\boldsymbol{mean([tmp]_{now-10s}^{now}) <_{0.95} 20}$
  - Assuming that tmp is a constant with normal distribution of observation error …
  - … the expected value of temperature computed from the data in last 10 seconds is less than 20 degrees with confidence 95%
- $\boldsymbol{lr([bat]_{now-10s}^{now}, now) <_{0.95} 20}$
  - Assuming that the observations have linear trend …
  - … the estimation of current battery level via linear regression calculated over the last 10 seconds interval is less than 20 with confidence 95%
- $\boldsymbol{lr([bat]_{now-60s}^{now}, now + 120s) >_{0.95} 20}$
  - Assuming the observations have linear trend and keep it …
  - … the estimation of the battery level in 120s from now calculated over the last 10 seconds interval is expected to be more than 20 with confidence 95%


```
1.   component role Cleaner
2.     [cleaning, charging,
              headingforchar
              waitingforclos
3.     int id
4.     int energyLevel
5.     bool inCharger
6.     Charger charger
7.     Area cleaningArea
8.     Camera[] cameras

9.     mode-switch-table
10.      below(energyLevel, ENERGYLOWLIMIT, 1min)
           && unset(charger) -> mode = findcharger
11.      below(energyLevel, ENERGYLOWLIMIT, 1min)
           && isset(charger) ->
                    mode = headingforcharger
12.      fabove(energyLevel, ENERGYLOWLIMIT, 1min, 2min)
           && isset(cleaningArea) -> mode = cleaning
13.      inCharger && below(energyLevel, ENERGYCHARGED)
           -> mode = charging
14.      unset(cleaningArea)
           && above(energyLevel, ENERGYLOWLIMIT)
           -> mode = waitingfordirt
15.      true -> mode = waitingforclosedirt
```

$\boldsymbol{below(x, y, w)}$

$$\Leftrightarrow lr([x]_{now-w}^{now}, now) <_{0.95} y$$

$\boldsymbol{above(x, y, w)}$

$$\Leftrightarrow lr([x]_{now-w}^{now}, now) >_{0.95} y$$

$\boldsymbol{fbelow(x, y, w, f)}$

$$\Leftrightarrow lr([x]_{now-w}^{now}, now + f) <_{0.95} y$$

$\boldsymbol{fabove(x, y, w, f)}$

$$\Leftrightarrow lr([x]_{now-w}^{now}, now + f) >_{0.95} y$$

Use in C++

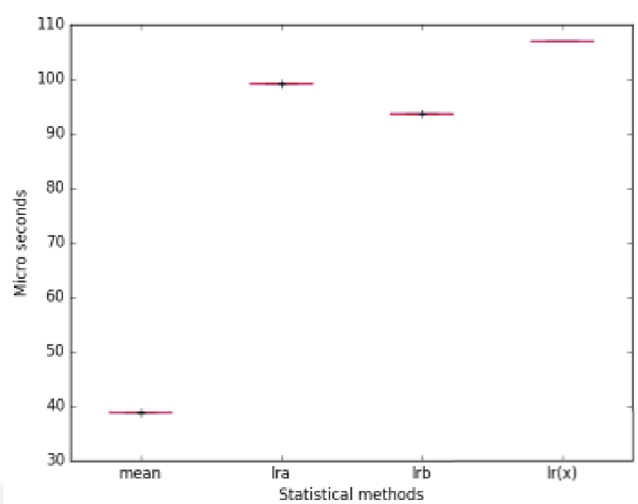
- Implementation as C++ library
- Optimized for embedded devices
  - Predictable memory footprint
  - No dynamic allocation

```
// Measurement part
TimeSeries<20, 6000> A;
// Measure data and keep history
Double data = sensor.measure();
A.addSample(data, time);
...

// Testing hypothesis over the mean of samples
StudentsDistribution m = A.getMean();
if ( m <= 20 )
  // ... switch mode as per specification
```


Feasibility on Embedded devices

- Question: Is the statistical approach to mode switching feasible (from performance perspective) on embedded devices?
- Platform: STM32F4-DISCOVERY embedded board
  - 168MHz, 192 KB RAM, 1MB flash memory



Conclusion & Open Challenges

- Reason about uncertainty in architecture mode-switching
- Backed by frequentists statistics
- Lightweight implementation, easy integration

- Limitations & Open Challenges
  - Requires error to be i.i.d. and to have normal distribution
  - Does not deal with outliers
    - Robust methods, e.g. quantile-based interpretation might help
  - Does not deal with uncertainty coming from human in the loop
    - Integrating of entities that are part of the system but not controlled – e.g. include human as regular component

Department of Distributed and Dependable Systems D3S

15

Ontological Reasoning as an Enabler of Contract-Based Co-Design

MPM4CPS COST Action Meeting

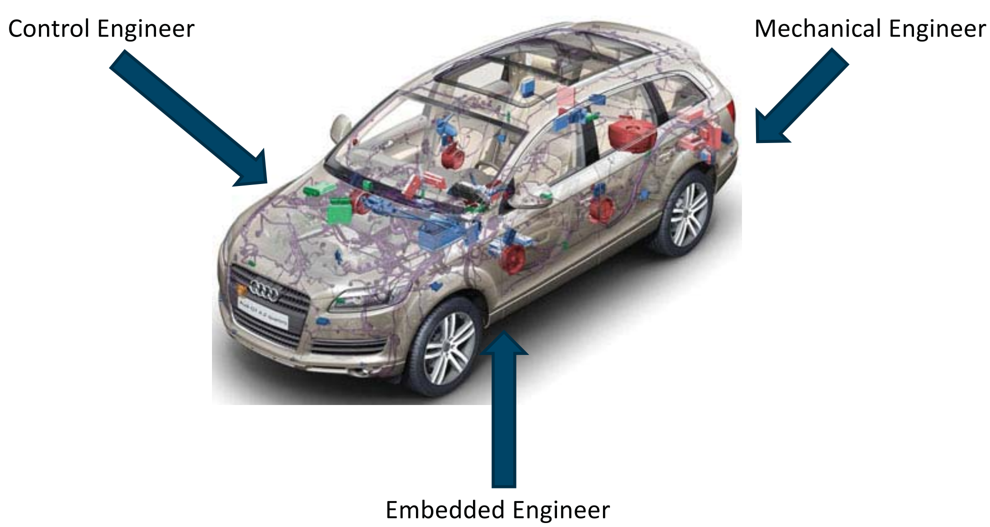
September 15th, 2016

Ken Vanherpen, Joachim Denil, István Dávid, Paul De Meulenaere, Pieter J. Mosterman, Martin Törngren, Ahsan Qamar, Hans Vangheluwe

CoSys-Lab Constrained Systems Lab University of Antwerp

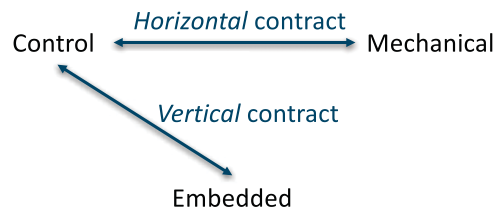
McGill UNIVERSITY

Problem Statement



Contract-Based Design – State-of-the-Art

Definition of a contract



$$C = (A, G)$$

Three operators

- Conjunction:
  $C1 \wedge C2 = (A, G)$ with: $A = A1 \cup A2$
  $G = G1 \cap G2$
- Composition:
  $C1 \otimes C2 = (A, G)$ with: $A = (A1 \cup A2) \cup \neg (G1 \cap G2)$
  $G = G1 \cap G2$
- Refinement:
  $C1 \preccurlyeq C2$ iff : $A2 \supseteq A1$
  $G2 \subseteq G2$

State of the art

A.Benveniste et al. Contracts for Systems Design : Theory. Tech. Rep. RR-8759, INRIA, 2015.

Elementary Requirements Power Window

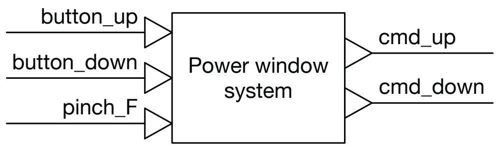
1. The power window should start moving within 200ms after a command is issued.

2. The power window shall be fully opened or closed within 4.5 s.

3. When closing the power window, a force of no more than 100N may be present.

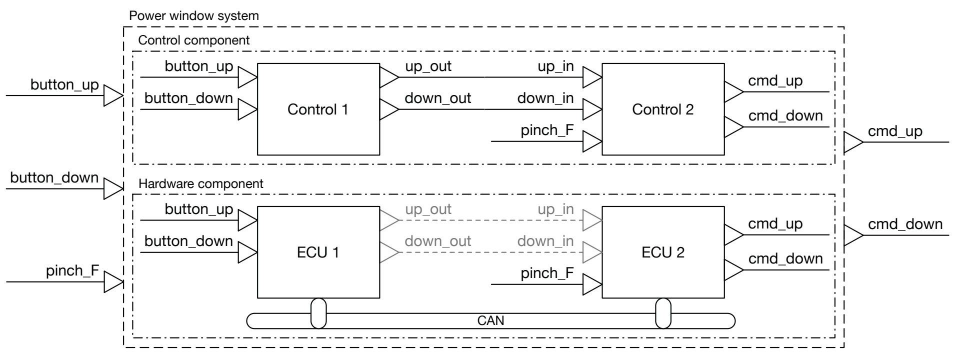
4. Detection of a clamped object when closing the window should lower the window by 10 cm.

System Contract Power Window

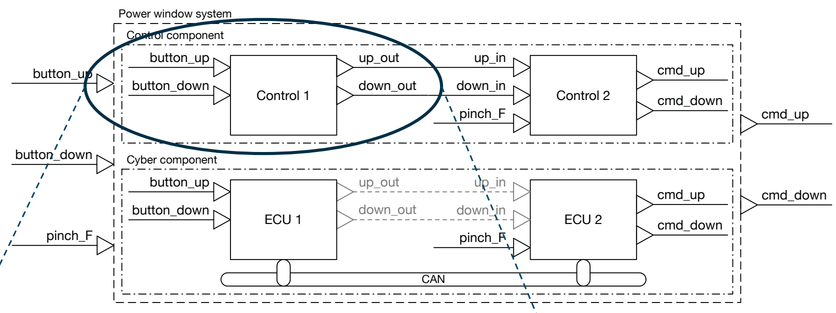


Assumptions	*pinch_F* will be lower than 1000 N. *button_up* occurs sporadic with a minimum period of 100 ms. *button_down* occurs sporadic with a minimum period of 100 ms
Guarantees	Delay between *button_up* and *cmd_up* within [0 ms, 200 ms]. Delay between *button_down* and *cmd_down* within [0 ms, 200 ms]. Maximum activation time *cmd_up* within [0 ms, 4.5 s]. Maximum activation time *cmd_down* within [0 ms, 4.5 s]. If *pinch_F* exceeds 100 N, delay between *pinch_F* and *cmd_down* within [0ms, 1ms]. If *pinch_F* exceeds 100 N, activation time *cmd_down* within [0,43 s, 0.43 s].

Decomposition of the Power Window System

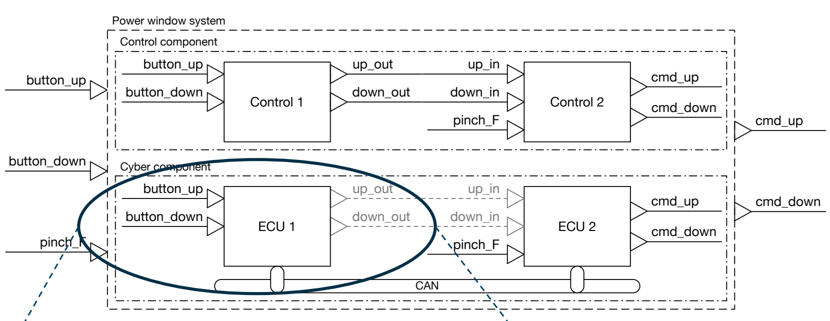


Applicability of the Current Methodologies on a Co-Design Engineering Problem



Assumptions	*button_up* occurs sporadic with a minimum period of 50 ms. *up_out* occurs sporadic with a minimum period of 2 ms.
Guarantees	Delay between *button_up* and *up_in* within [0 ms, 52 ms].

Applicability of the Current Methodologies on a Co-Design Engineering Problem



Assumptions	*button_up* occurs sporadic with a minimum period of 40 ms. *Runnable#actuation* occurs each 40 ms. Delay between *Runnable#actuation* and *up_out* within [0 ms, 10 ms].
Guarantees	whenever *button_up* occurs, *up_out* occurs. Timer occurs each 10 ms. Delay between *button_up* and up_*out* within [200 us, 10 ms + 1.3 ms].

Applicability of the Current Methodologies on a Co-Design Engineering Problem

Shortcomings:

- Domain contract contains properties on which the domain engineer lacks the ability to reason about
- No clear separation between what is assumed from the other domain(s) and what should be guaranteed under these conditions

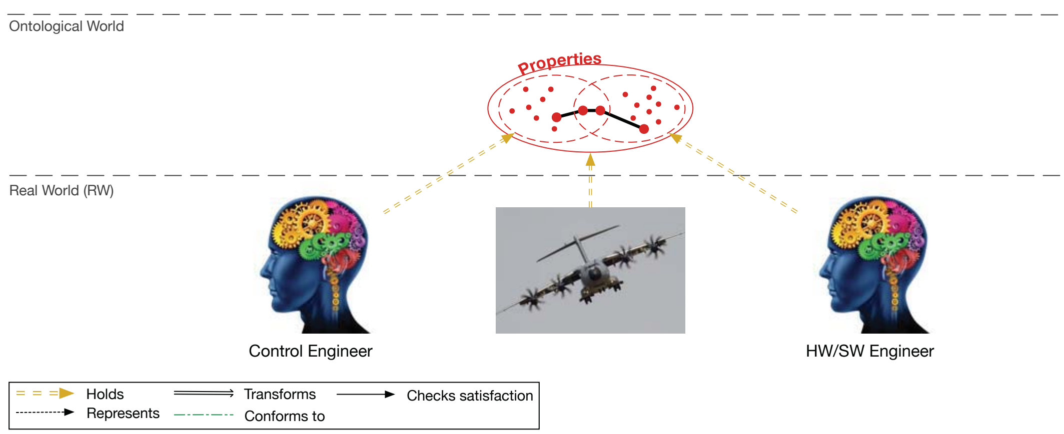
A Contract-Based Co-Design Methodology

Based on Ontological Reasoning:

K. Vanherpen et al. Ontological Reasoning for Consistency in the Design of Cyber-Physical Systems. CPPS 2016.

A Contract-Based Co-Design Methodology

Based on Ontological Reasoning:

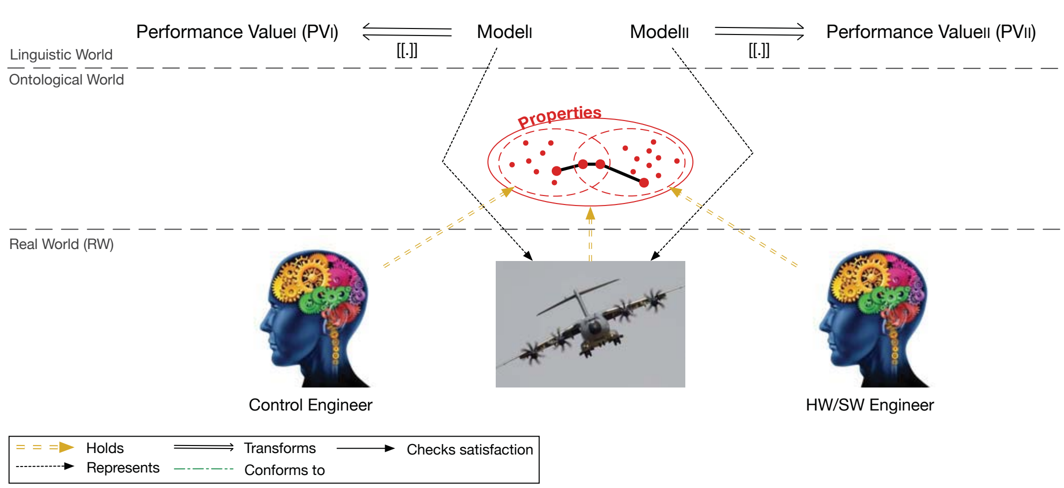


K. Vanherpen et al. Ontological Reasoning for Consistency in the Design of Cyber-Physical Systems. CPPS 2016.

11

A Contract-Based Co-Design Methodology

Based on Ontological Reasoning:

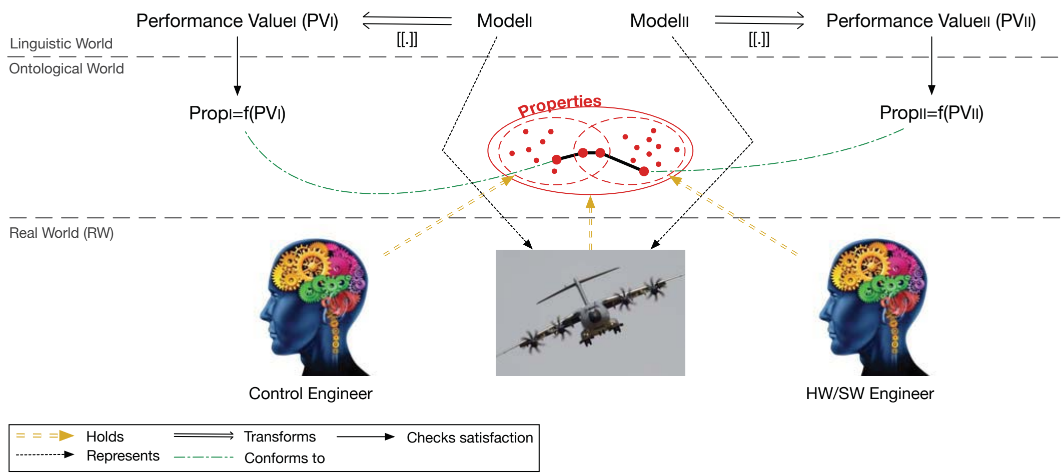


K. Vanherpen et al. Ontological Reasoning for Consistency in the Design of Cyber-Physical Systems. CPPS 2016.

12

A Contract-Based Co-Design Methodology

Based on Ontological Reasoning:



K. Vanherpen et al. Ontological Reasoning for Consistency in the Design of Cyber-Physical Systems. CPPS 2016.

A Contract-Based Co-Design Methodology

Phase 1 – Negotiation

Control

Mapping

Assumptions	Processor Clock = 1 MHz # instr Comp1 = 200 200 us ≥ Periodicity Comp1 ≤ 100 ms Min interval inputs = 100 ms Max comm time = 50 ms Processor Clock = 8 MHz # instr Comp2 = 1000 250 us ≥ Periodicity Comp2 ≤ 750 us
Guarantees	Max E-E Latency Comp1 = 199 ms Max E-E Latency Comp2 = 1 ms

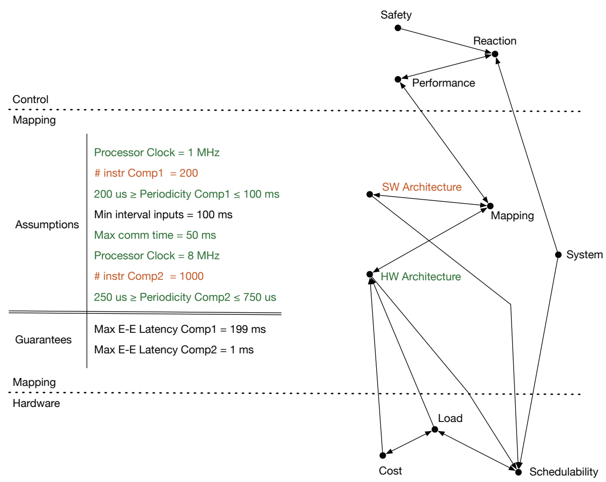
Mapping

Hardware

K. Vanherpen et al. Ontological Reasoning as an Enabler of Contract-Based Co-Design. CyPhy 2016.

A Contract-Based Co-Design Methodology

Phase 1 – Negotiation

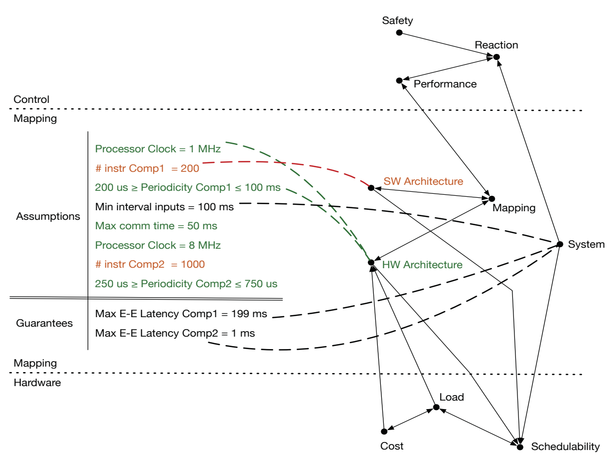


K. Vanherpen et al. Ontological Reasoning as an Enabler of Contract-Based Co-Design. CyPhy 2016.

15

A Contract-Based Co-Design Methodology

Phase 1 – Negotiation

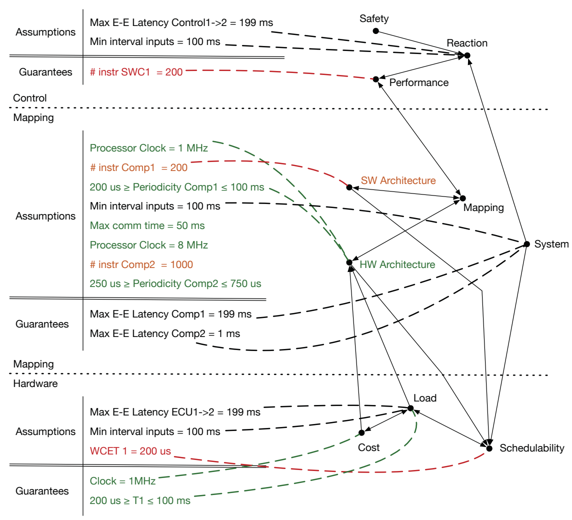


K. Vanherpen et al. Ontological Reasoning as an Enabler of Contract-Based Co-Design. CyPhy 2016.

16

A Contract-Based Co-Design Methodology

Phase 2 – Deriving the domain contracts

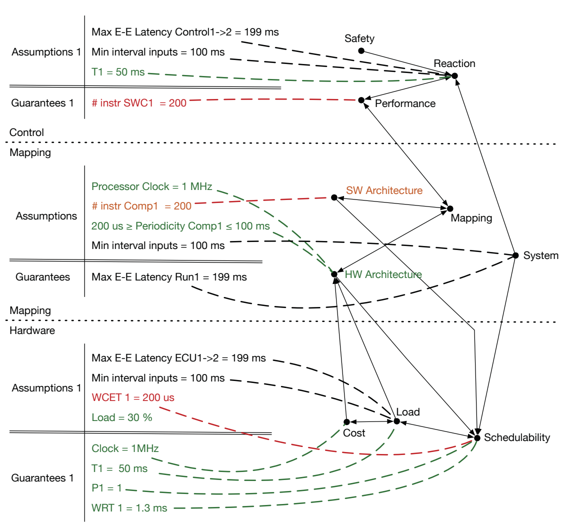


K. Vanherpen et al. Ontological Reasoning as an Enabler of Contract-Based Co-Design. CyPhy 2016.

17

A Contract-Based Co-Design Methodology

Phase 3 – Refinement of the domain contracts



K. Vanherpen et al. Ontological Reasoning as an Enabler of Contract-Based Co-Design. CyPhy 2016.

18

Thank you

Ing. Ken Vanherpen | ken.vanherpen@uantwerp.be

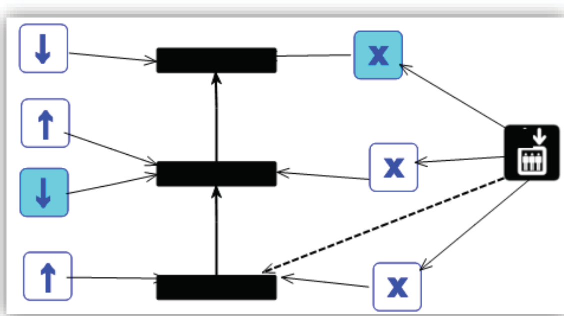
19

Verification of Domain-Specific Models with ProMoBox

Bart Meyers
Universiteit Antwerpen
bart.meyers@uantwerpen.be

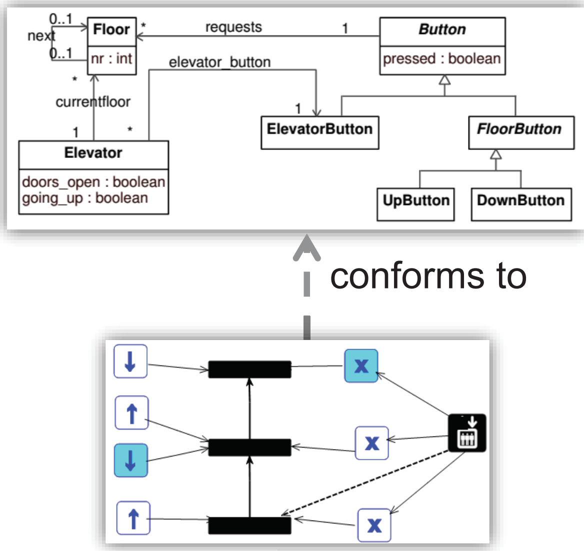
Intro: Domain-Specific Modelling

- Modelling of complex systems for **domain users**
  - Familiar domain **concepts**
  - Incorporate domain **constraints**
- **Precise** and **well-defined** models
- Example: elevator system

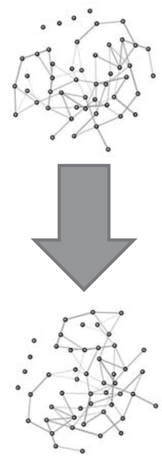


Intro: Language Engineering

– Meta-modeling

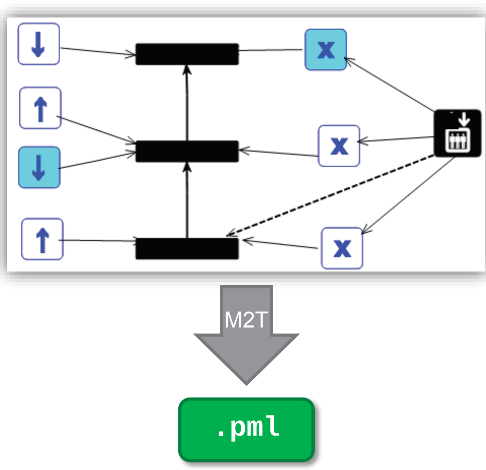


– Model Transformation



Properties for DSMLs: State of the Art

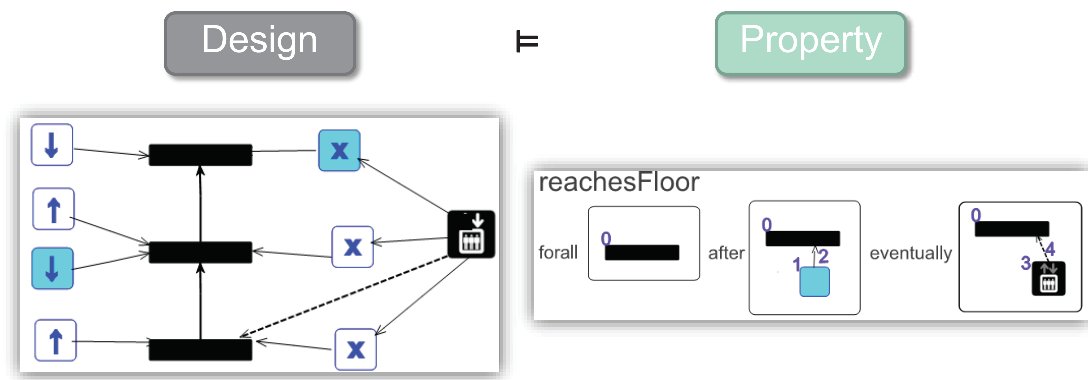
Design ⊨ Property



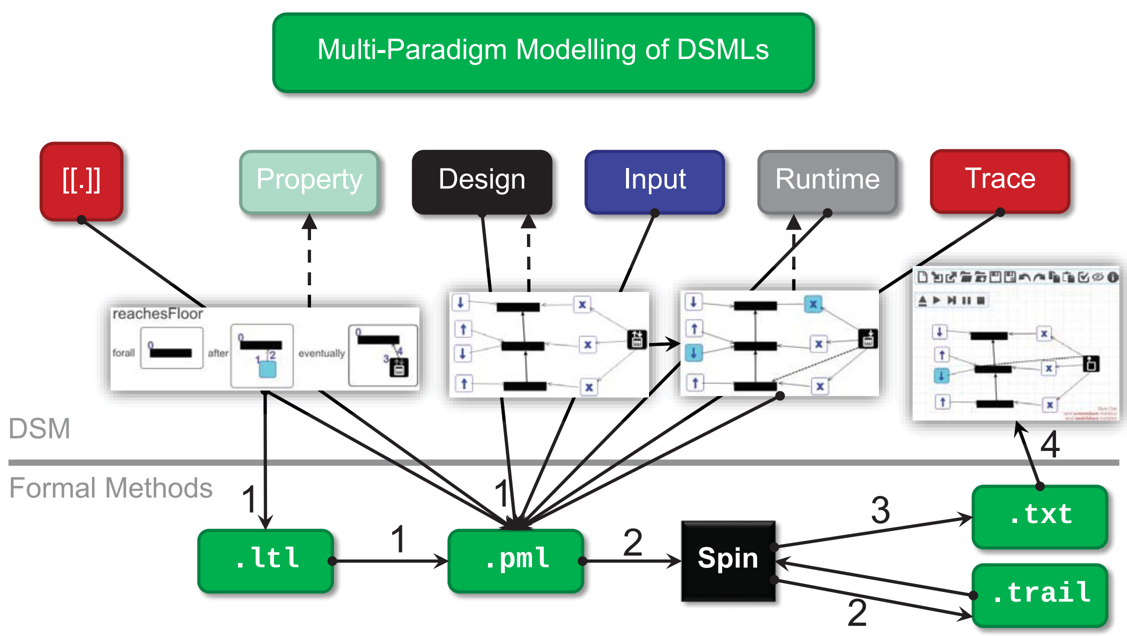
⊨

$$\Box(((go0\wedge up0)\vee\Diamond(floor0\vee idle))\rightarrow((\neg(floor0)\vee\neg(floor0\vee idle))\mathcal{U}((floor0\vee idle)\wedge(((floor0)\vee\neg(floor0\vee idle))\mathcal{U}((floor0\vee idle)\wedge((\neg(floor0)\vee\neg(floor0\vee idle))\mathcal{U}((floor0\vee idle)\wedge(((floor0)\vee\neg(floor0\vee idle))\mathcal{U}((floor0\vee idle)\wedge(\neg(floor0)\mathcal{U}(floor0\vee idle)))))))))))\vee\Box(((go1\wedge up1\wedge down1)\vee\Diamond(floor1\vee idle))\rightarrow((\neg(floor1)\vee\neg(floor1\vee idle))\mathcal{U}((floor1\vee idle)\wedge(((floor1)\vee\neg(floor1\vee idle))\mathcal{U}((floor1\vee idle)\wedge((\neg(floor1)\vee\neg(floor1\vee idle))\mathcal{U}((floor1\vee idle)\wedge(((floor1)\vee\neg(floor1\vee idle))\mathcal{U}((floor1\vee idle)\wedge(\neg(floor1)\mathcal{U}(floor1\vee idle)))))))))))\vee\Box(((go2\wedge down2)\vee\Diamond(floor2\vee idle))\rightarrow((\neg(floor2)\vee\neg(floor2\vee idle))\mathcal{U}((floor2\vee idle)\wedge(((floor2)\vee\neg(floor2\vee idle))\mathcal{U}((floor2\vee idle)\wedge((\neg(floor2)\vee\neg(floor2\vee idle))\mathcal{U}((floor2\vee idle)\wedge(((floor2)\vee\neg(floor2\vee idle))\mathcal{U}((floor2\vee idle)\wedge(\neg(floor2)\mathcal{U}(floor2\vee idle))))))))))$$

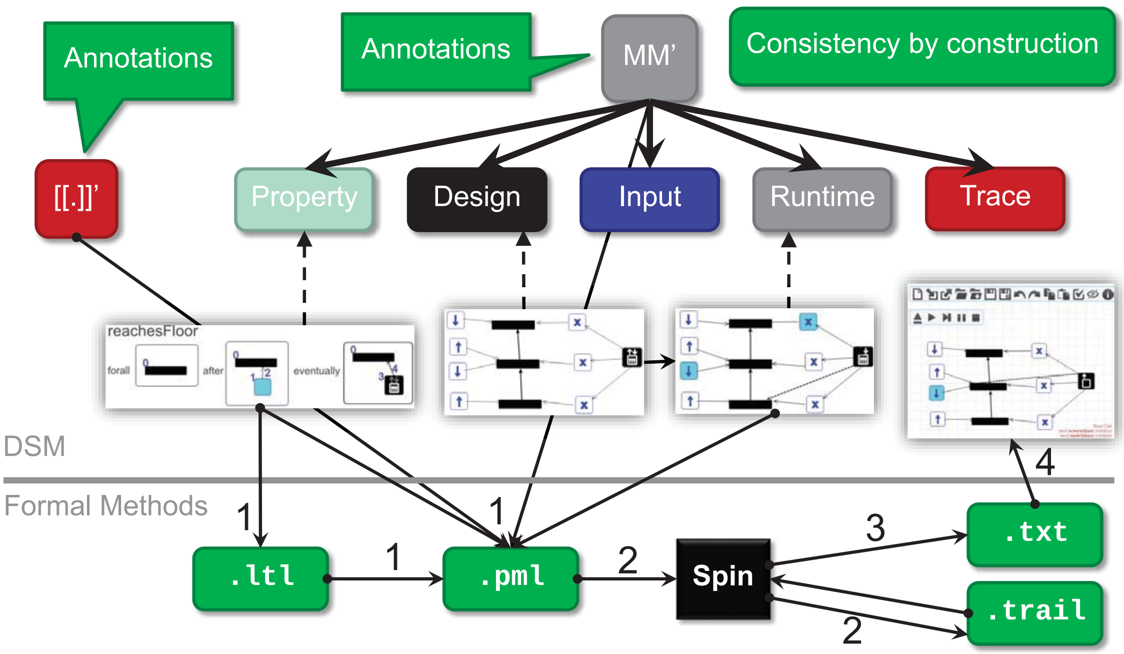
Properties for DSMLs: Property DSML



Properties for DSMLs: Five Languages

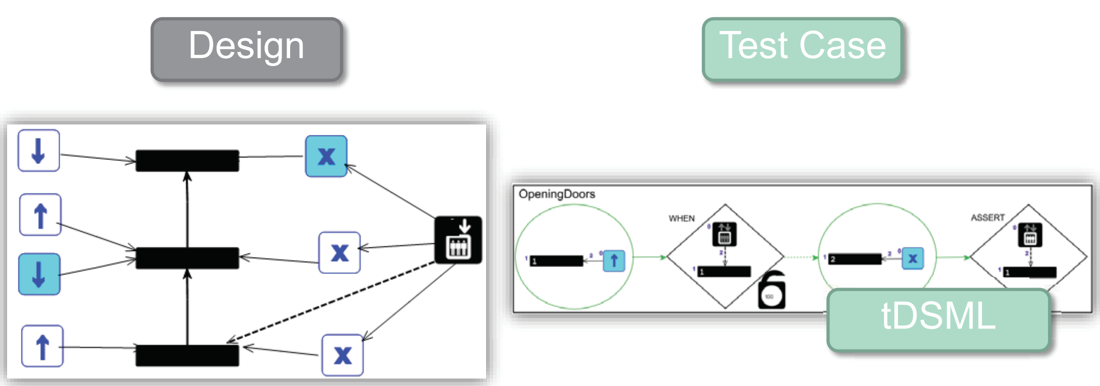


Properties for DSMLs: Consistency



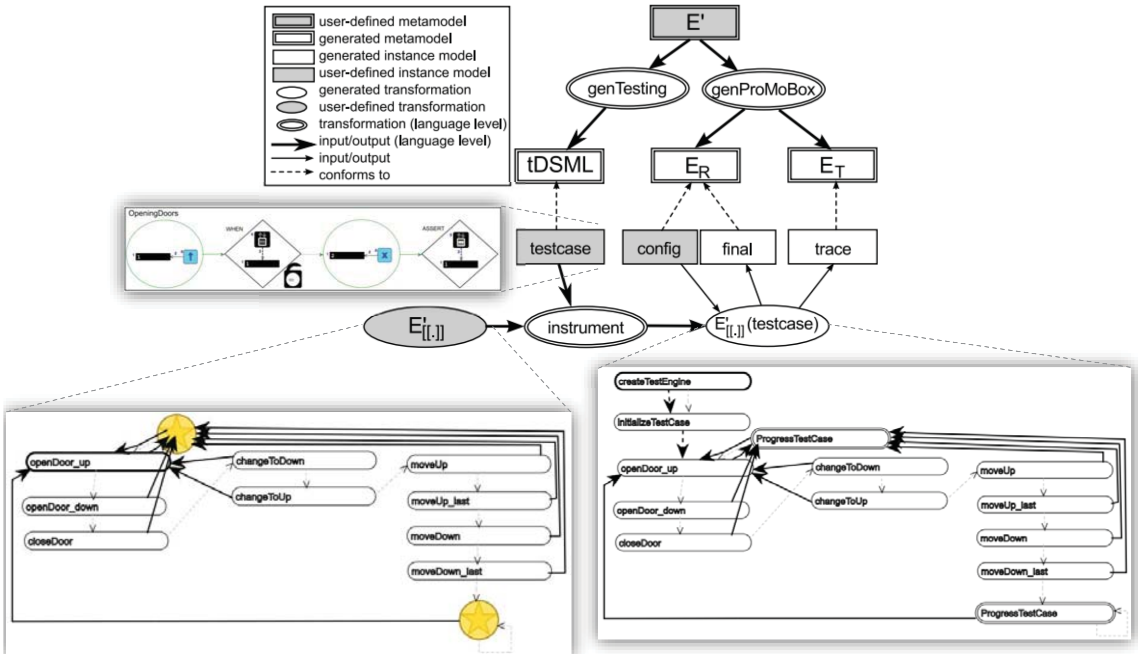
McGill School of Computer Science nessis 7 FLANDERS MAKE Ansymo Antwerp Systems & Software Modelling University of Antwerp

Properties for DSMLs: Testing



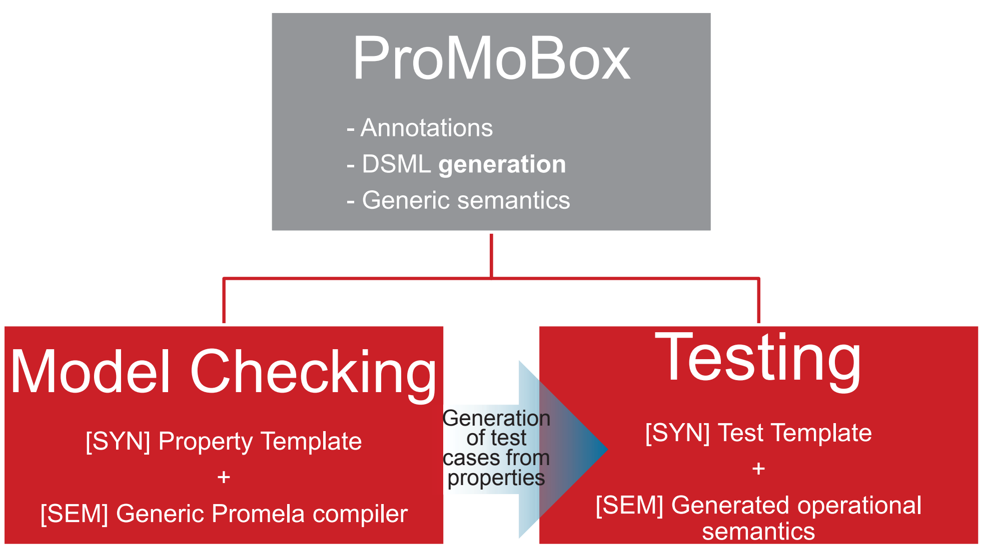
McGill School of Computer Science nessis 8 FLANDERS MAKE Ansymo Antwerp Systems & Software Modelling University of Antwerp

Properties for DSMLs: Testing (Approach)



McGill School of Computer Science nessis 9 FLANDERS MAKE Ansymo Antwerp Systems & Software Modelling University of Antwerp

Conclusion and future



McGill School of Computer Science nessis 10 FLANDERS MAKE Ansymo Antwerp Systems & Software Modelling University of Antwerp

Publications

- Bart Meyers, Joachim Denil, Istvan David and Hans Vangheluwe. Automated Testing Support for Reactive Domain-Specific Modelling Languages. Submitted to International Conference on Software Language Engineering (SLE '16), 2016.
- Bart Meyers and Hans Vangheluwe. Modelling Language Engineering to Include Temporal Properties in Domain-Specific Modelling. Submitted to Transactions on Software Engineering, 2015.
- Romuald Deshayes, Bart Meyers, Tom Mens and Hans Vangheluwe. ProMoBox in Practice : A Case Study on the GISMO Domain-Specific Modelling Language. In "Proceedings of the 8th Workshop on Multi-Paradigm Modeling (MPM 2014)", CEUR Workshop Proceedings, vol. 1237, p. 21-30, 2014.
- Bart Meyers and Hans Vangheluwe. A Multi-Paradigm Modelling Approach for the Engineering of Modelling Languages. In "Proceedings of the Doctoral Symposium of the ACM/IEEE 17th International Conference on Model Driven Engineering Languages and Systems", CEUR Workshop Proceedings, vol. 1321, p. 1-8, 2014.
- Bart Meyers, Romuald Deshayes, Levi Lucio, Eugene Syriani, Manuel Wimmer and Hans Vangheluwe. ProMoBox: A Framework for Generating Domain-Specific Property Languages. In "Proceedings of the 7th International Conference on Software Languages Engineering (SLE 2014)", Lecture Notes on Computer Science, vol. 8706, p. 1-20, 2014.
- Bart Meyers, Manuel Wimmer, and Hans Vangheluwe. Towards Domain-specific Property Languages: The ProMoBox Approach. In "Proceedings of the 2013 ACM Workshop on Domain-specific Modeling", p. 39-44, ACM New York, NY, USA, 2013.

Semantic-aided enterprise application integration

Gdańsk, 2016

About us

Željko Vuković

zeljkov@uns.ac.rs

- University of Novi Sad
  - Faculty of Technical Sciences
    - Chair of Informatics

Nikola Milanović

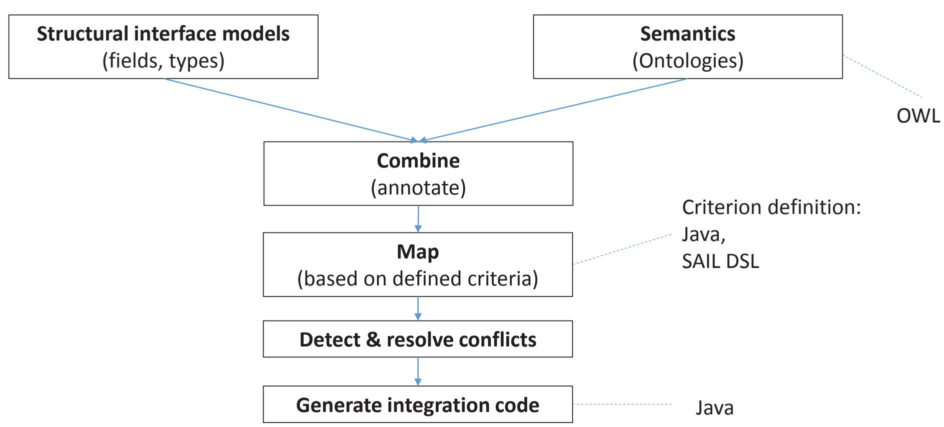
- OPTIMAL SYSTEMS GmbH
  - Berlin

Our goal

- Automate or semi-automate:
  - Interface mapping in enterprise application integration
  - Detection and resolution of semantic conflicts

- Enterprise application integration
  - Persuading into cooperation things that were not originally meant to work together

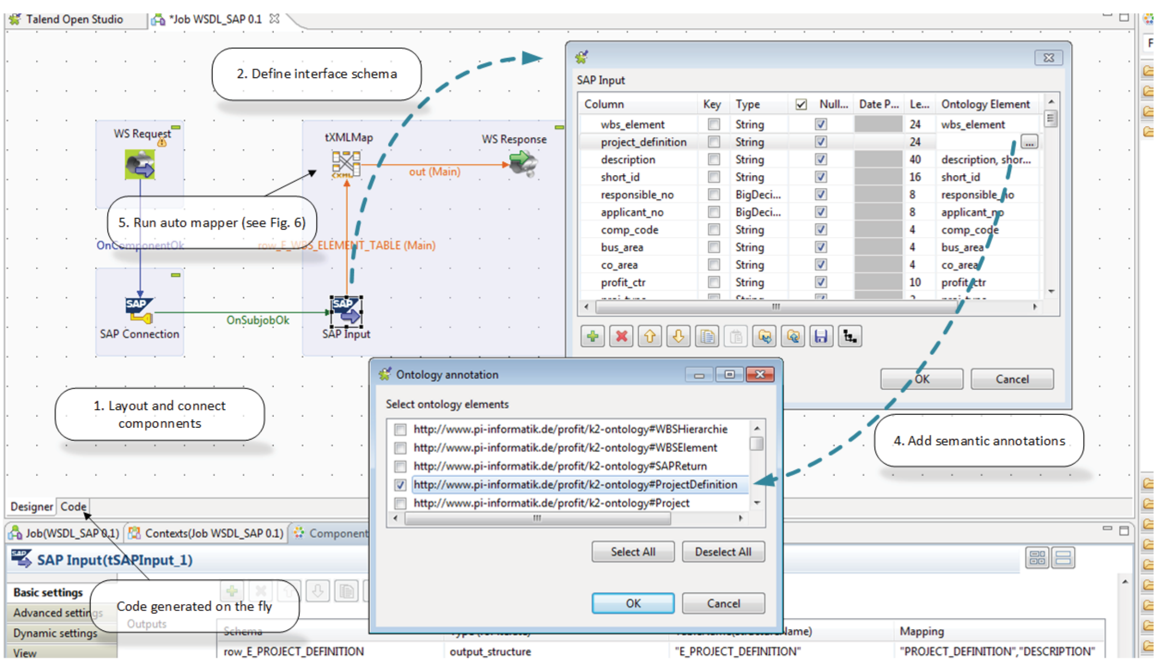
Approach

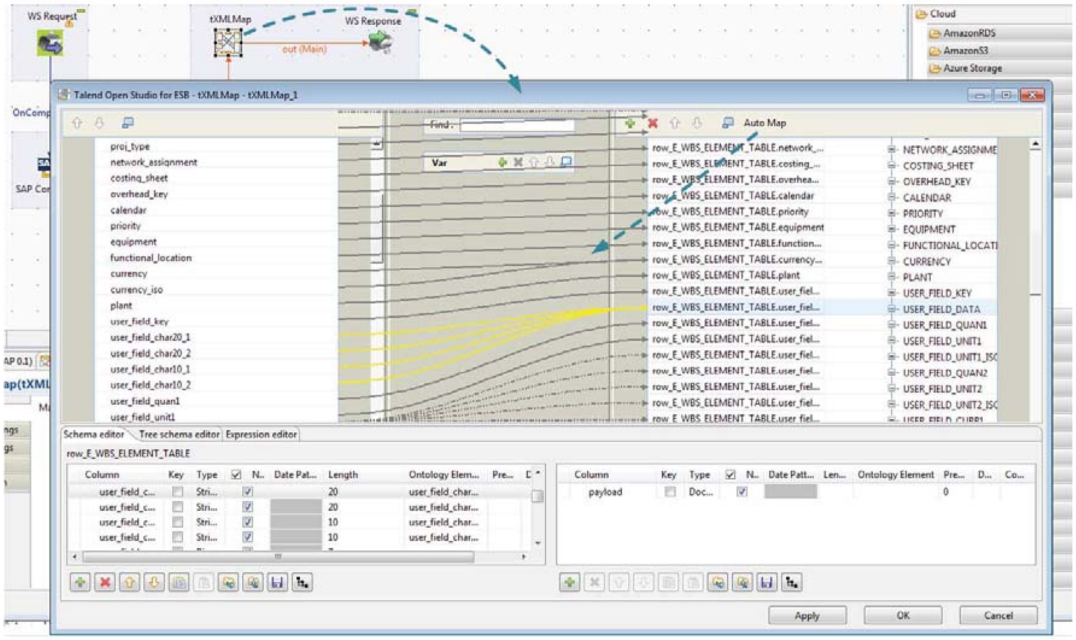


Implementation

- Mod of Talend Open Studio
  www.talend.com/products/talend-open-studio

  - plenty of connectors available (databases, flat files, web services, …)
  - can model processes, interaction
  - extendable
  - open-source





How we relate?

- If physical things have a cyber interface, we can integrate them
  - Fire alarm? — 2001:0db8:85a3:0000:0000:8a2e:0370:7334
    - Call fire brigade — 112
    - Cut electrical power — 2001:db8:a0b:12f0::1
- Formalisms
  - Talend, UML, XML, OWL, Java, SAIL DSL

Published work

- Vuković, Željko, et al. "SAIL: A Domain-Specific Language for Semantic-Aided Automation of Interface Mapping in Enterprise Integration." *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer International Publishing, 2015.
- Vuković, Ž., Milanović, N., Vaderna, R., Dejanović, I., Milosavljević, G. and Malbaša, V., Semantic-aided automation of interface mapping in enterprise integration with conflict detection. *Information Systems and e-Business Management*, pp.1-18.; 2016

On Interoperability of IoT platforms

SRIPAS

Introduction

- INTER-IoT – one of seven projects that started on January 1, 2016
- Silo problem in IoT
  - multiple incompatible platforms / systems / applications → IoT artifacts
  - vendors do not have incentive to make their "solutions" collaborate
    - large companies pretend, but fight for territory
    - SME's do not want / do not have resources
- EC believes that having interoperability in IoT world will be beneficial to make EU more competitive

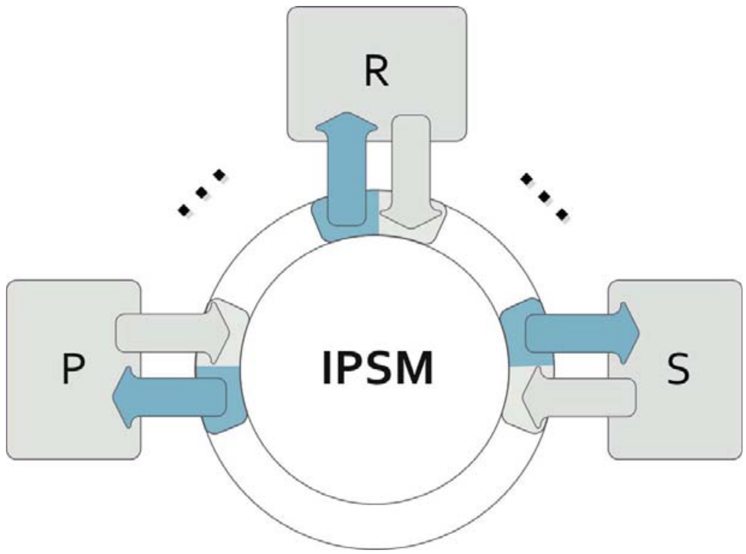
Use cases

- Transport and logistics
  - Port of Valencia
    - different “IoT platforms”
  - outsiders – truck companies
    - IoT-like platforms
  - need to make them talk to each other
- (e/m)Health
  - Body-net type solution
  - Internet enabled devices in homes
  - doctor needs a ”unified view”

What are we to do?

- Interoperabilily on all levels of software stack
- Methodology for making IoT artifacts interoperable
- “We” are interested in semantic interoperability
- Assumptions
  - more than **two** IoT artifacts to connect / collaborate
  - these artifacts may (later) become involved in other collaborations
  - focus on information exchange (messages)
  - individual artifacts should not(!) change

IoT Platform Semantic Mediator **(IPSM)**



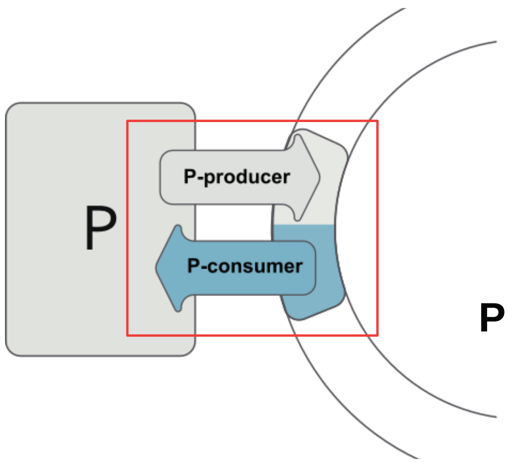
Data & Semantics interoperability **(semantics via ontologies)**

- High-level "core" ontology – **GOIoTP**
  - possibly based on the *OpenIoT* ontology
  - to be developed in parallel with use-cases

- Platform-specific ontologies
  - capture platform (domain)-specific semantics
  - might be provided by the platform or extracted by the *platform integrator*

Existence of a joint ontology is **not** assumed

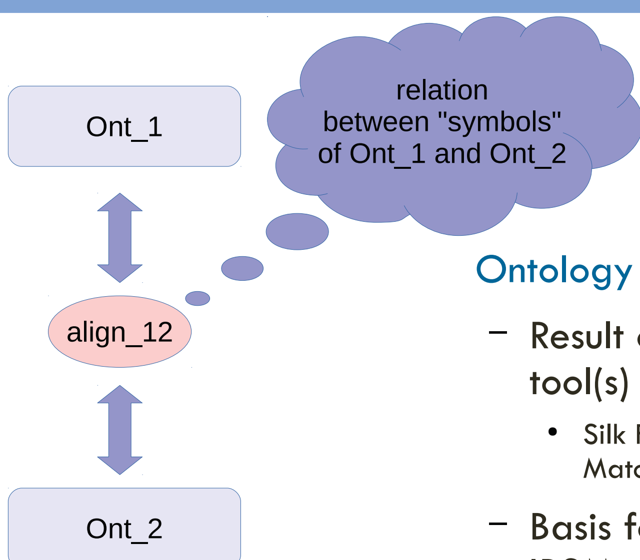
IoT Platform Semantic Mediator (**platform translator**)



P *translator*

P data ↔ **P** ontology

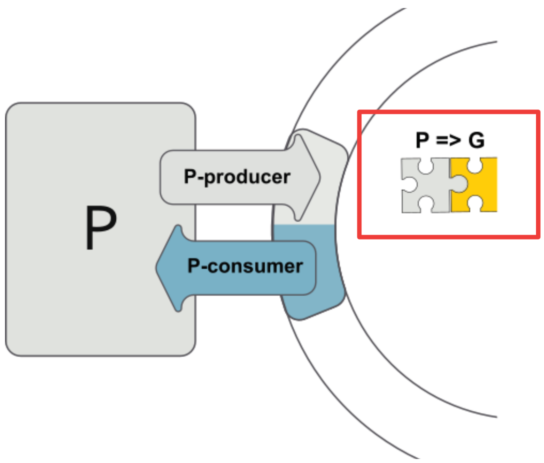
Data & Semantics interoperability (**ontology alignments**)



Ontology alignment

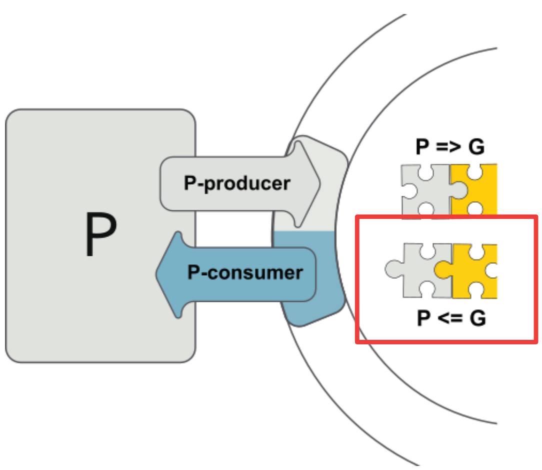
- Result of applying matching tool(s)
  - Silk Framework, COMA, LogMap, S-Match, …
- Basis for semantic *conversion in* IPSM

IoT Platform Semantic Mediator
(**platform ontology alignments**)



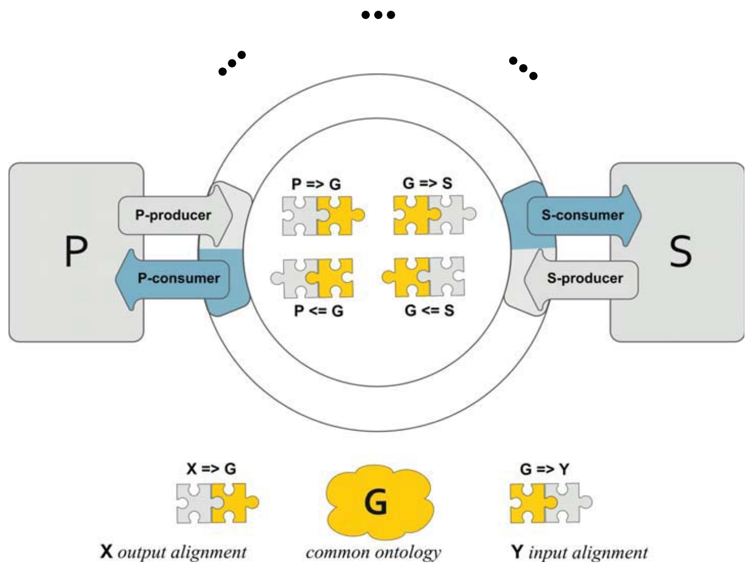
P *output alignment*

IoT Platform Semantic Mediator
(**platform ontology alignments**)



P *input alignment*

IoT Platform Semantic Mediator (**architecture overview**)



Solution – methodology outline

- Each artifact has implicit/explicit semantics
  - make semantics explicit via OWL ontology
  - complexity of this task depends on
    - data type we work with
    - quality of documentation
- Develop domain specific ontologies (together with the core this results in the common ontology G)

Solution – methodology outline (cntd.)

- Align *artifact ontologies* (do not merge) with the *common ontology* **G** and store the alignments (for translations)
- Develop *translators* from individual artifacts (e.g. XML message) to OWL representation
  - no need to make changes in the systems
- Instantiate the IPSM – one IPSM for each group of artifacts to collaborate
- Adding a new artifact
  - do the same as above (potentially with new ontology module)

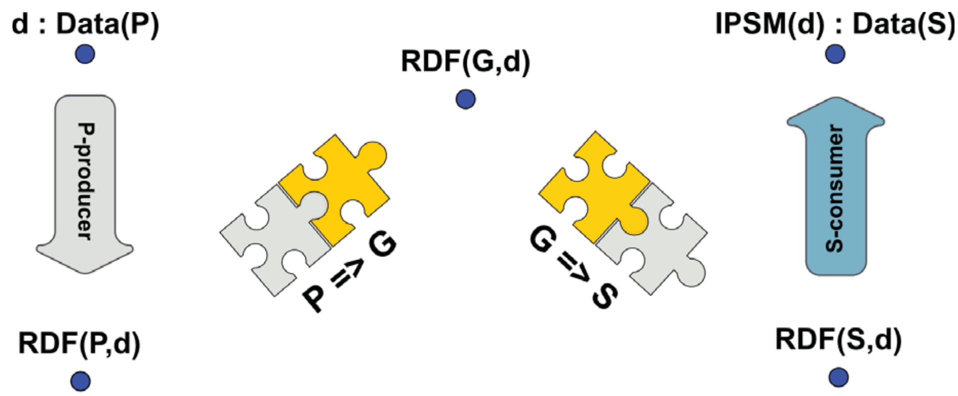
PROBLEM SOLVED

☺

IoT Platform Semantic Mediator (**“instance ontology” G**)

- Built upon the “core” ontology GOIoTP
- Can be extended in a *modular* way
- Modularity benefits
  - platform translator/alignments don't have to be extended
  - platform alignments can also be extended in a conservative way

IoT Platform Semantic Mediator (**semantic data conversion**)



IoT Platform Semantic Mediator (**semantic data conversion**)

- uses:
  - platform translator(s)

    platform *data* ↔ platform *semantics* in RDF(+)
  - platform conversion alignments

    platform semantics ↔ common semantics
- RDF(+) — semantic data with *IPSM-specific metadata*, e.g., *routing* information

IoT Platform Semantic Mediator (**platform requirements**)

IPSM **requirements** for the IoT platforms that are to cooperate:

- explicitly documented meaning, formats, structure description (e.g. OWL/RDF ontologies, database/XML/JSON schemas) of data to be interchanged
- explicitly documented description of services and their invocation methods (e.g. using WSDL)

IoT Platform Semantic Mediator (**voluntary interoperability**)

- voluntary cooperation between IoT platforms
  - instantiation of IPSM between them
- voluntary joining a "fedaration of IoT platforms"
  - adjustment of common semantics used by the IPSM instance used by the federation
  - instantiation of *translators* (data and semantics)

Several Issues for Modelling, Implementation and Control of Cyber-Physical Systems

Oksana Nikiforova, Andrejs Romanovs
Nadezhda Kunicina, Anatolijs Zabasta

Riga Technical University

EUROPEAN COOPERATION IN SCIENCE AND TECHNOLOGY

Oksana Nikiforova, professor, Dr.sc.ing.,
Riga Technical University
Faculty of Computer Science and Information Technology,
Institute of Applied Computer Systems,
Department of Applied Computer Science,
1 Setas Str., office 510, Riga, Latvia
Contacts: +371 67089598
oksana.nikiforova@rtu.lv

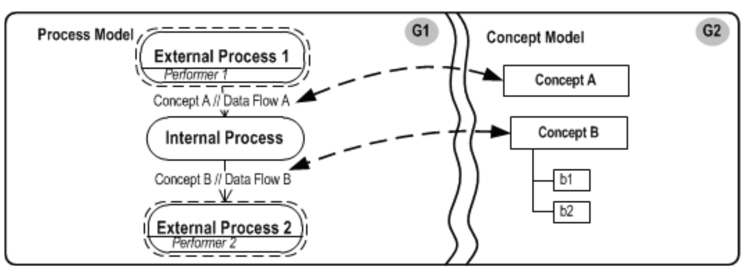
2

Application of the Two-Hemisphere Model for Modelling of CPS

- Cyber-Physical Systems (CPS) are heterogeneous systems, which offer
  - reliability, performance and robustness to allow companies to be competitive.
- To cope with the complexity of the execution of such, it is necessary to define an approach to tame its complexity.
- This approach should be flexible and generic in order to adapt to any type of component of such system and thus, should offer an ability to manage system integrity.
- The Two-hemisphere model driven (2HMD) approach has been successfully applied for domain modelling and software design and is applicable for both human understanding and automatic transformations.
  - Can be applied for modelling of such complex system as CPS, where components of CPS may be considered as a conceptual classes, which preform the particular operations and meet the defined requirements
  - The requirements for different components can be presented by separate process models under the unified conceptual class model
  - 2HMD approach can help to identify conflict situations, where the additional analysis is required to share responsibilities between system components.

3

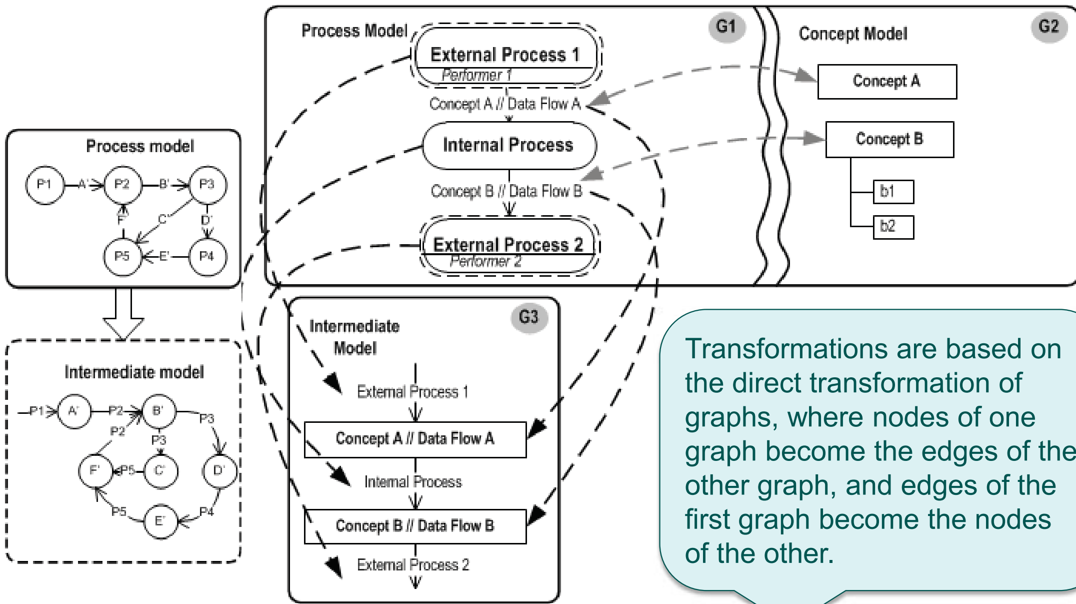
Two-hemisphere model based transformations



- The two-hemisphere model consists of:
  - Process diagram presents steps of system behaviour (or scenario) and defines
    - internal processes of the system enclosed by external processes performed by a set of performers.
    - data flow coming from one process to another, «structurally» defined by a conceptual class, called concept.
  - Concept diagram presents conceptual classes of the system and is similar to ER diagram without relationships.
- The model can have one or many process diagrams (scenarios) and only one (general for the whole system) concept diagram.

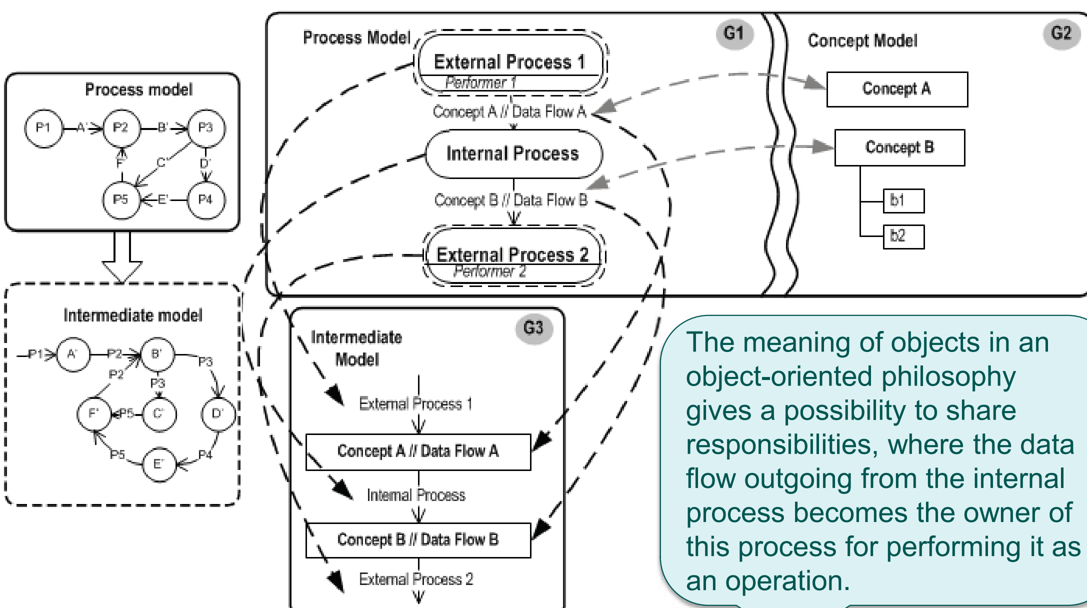
4

Two-hemisphere model based transformations



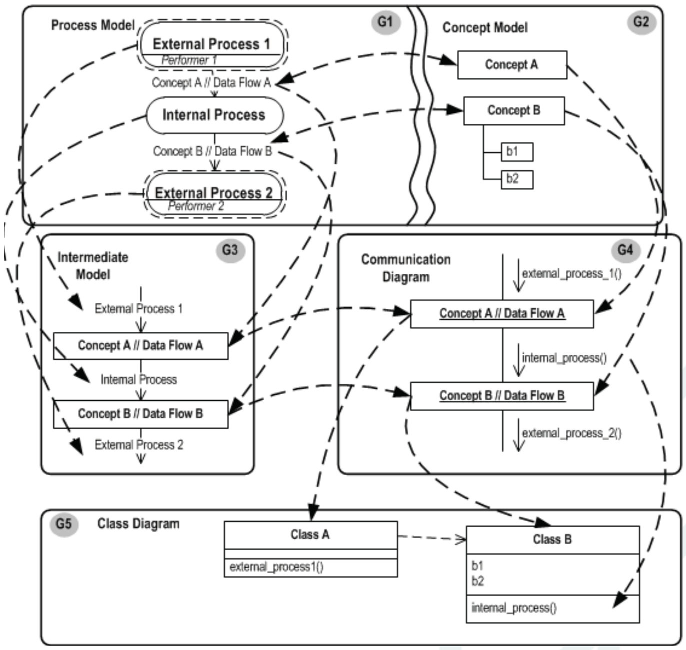
Transformations are based on the direct transformation of graphs, where nodes of one graph become the edges of the other graph, and edges of the first graph become the nodes of the other.

Two-hemisphere model based transformations



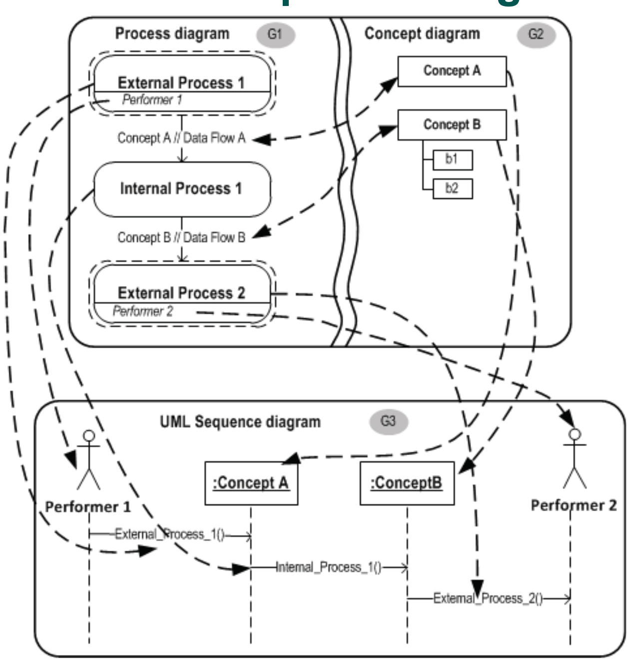
The meaning of objects in an object-oriented philosophy gives a possibility to share responsibilities, where the data flow outgoing from the internal process becomes the owner of this process for performing it as an operation.

Two-hemisphere model based transformations for UML class diagram



7

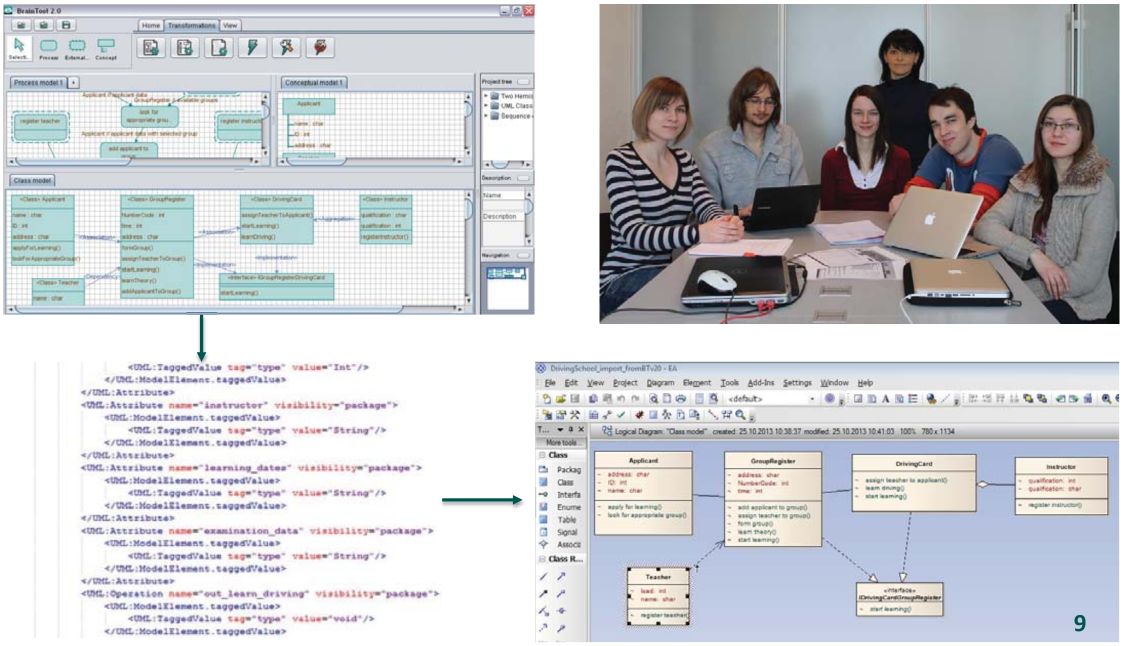
Two-hemisphere model based transformations for UML sequence diagram



- The same principle is used:
  - Processes are transformed into messages.
  - Concepts help to determine senders and receivers.
  - Performers assigned to external processes become actors.

8

BrainTool to support Two-Hemisphere Model Driven Approach



Andrejs Romanovs, Dr.sc.ing., MBA,
associate professor, leading researcher
Riga Technical University
Faculty of Computer Science and Information Technology,
Information Technology Institute,
2 Daugavgrivas Str., office 429, Riga, Latvia,
Contacts: +371 29637251
andrejs.romanovs@rtu.lv

10

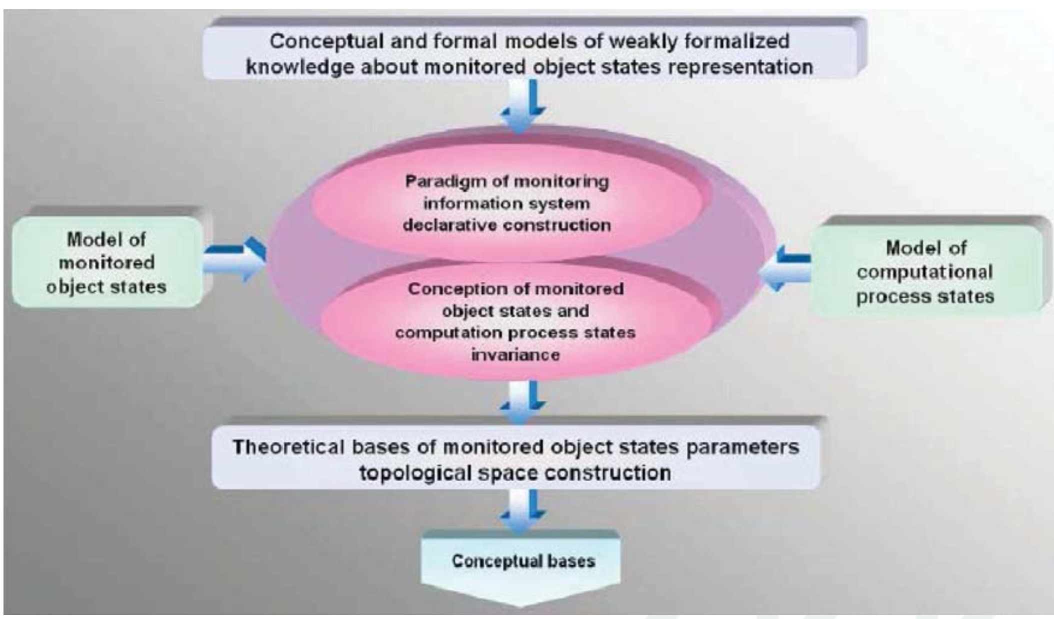
INFROM IT Technology Design and Development

OBJECTIVES:

- To develop methods for natural-technological systems (NTS) integrated modelling and simulation including reconfiguration of these systems under degradation process of their structures
- To develop methods for information representation under conditions of dynamics structure and data uncertainty
- To develop model and method of NTS monitoring and control systems dynamic reconfiguration
- To develop an innovative information technology and a complex cyber-physical system for analysis and synthesis of an integrated intelligent platform for NTS monitoring and control based on integrating heterogeneous information received from space and ground-based sensors, facilities, other industrial systems and individuals

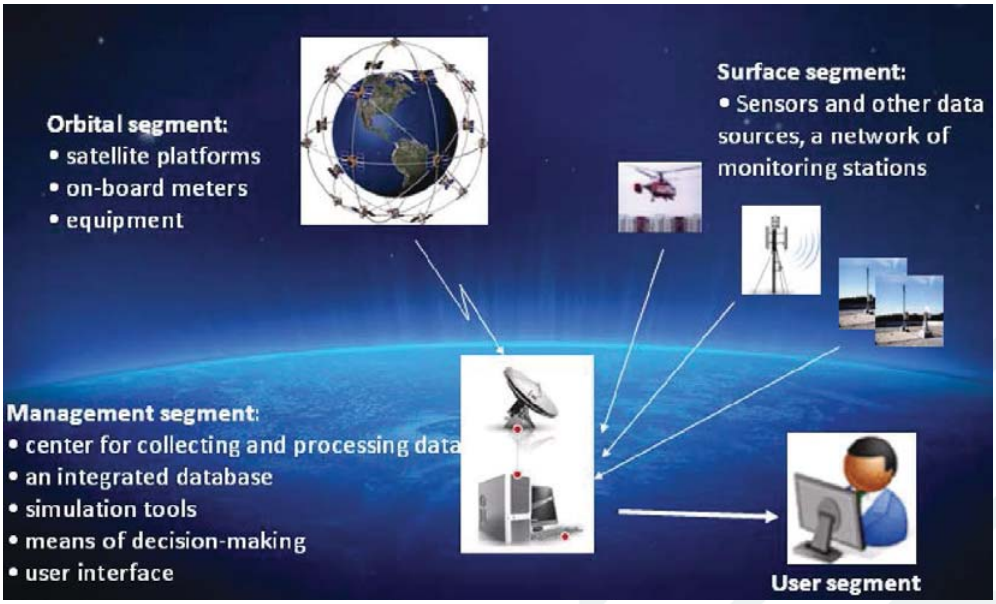
11

Fundamental scientific base



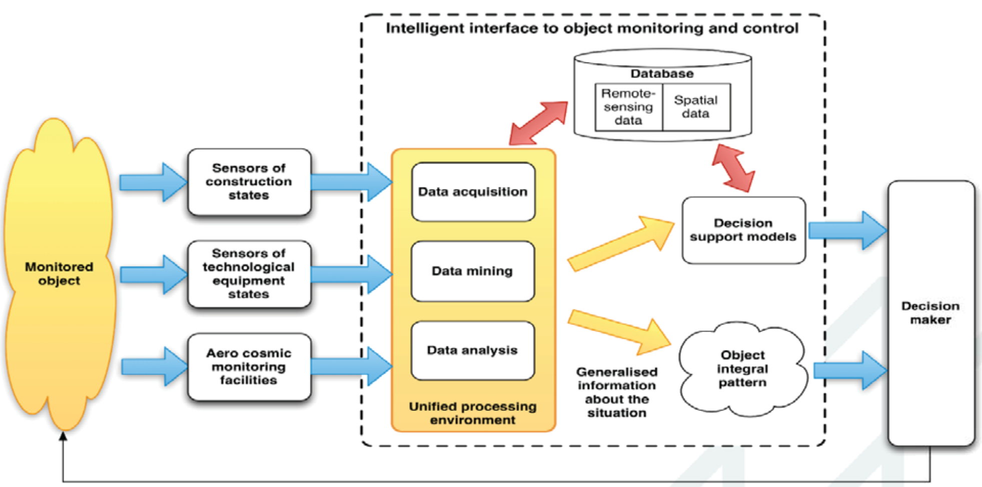
12

Structural model of INFROM intelligent technology



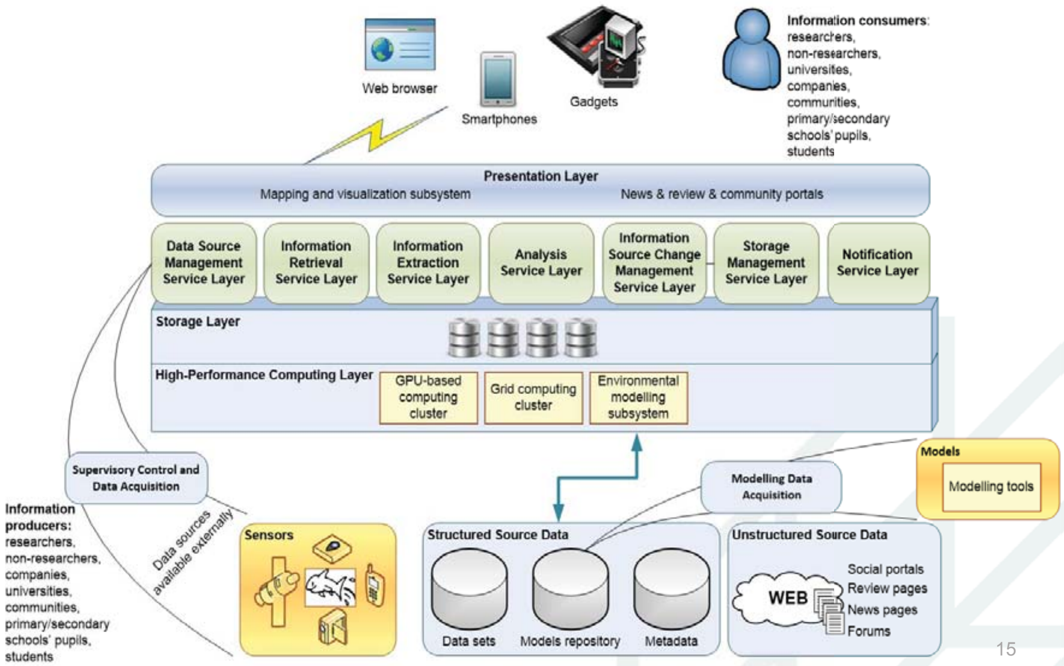
13

Framework for integrated remote sensing and monitoring



14

High-Level System Architecture

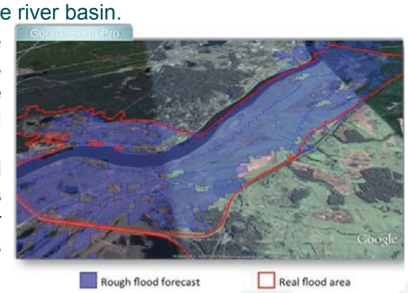


Demonstration case and real-time experiments

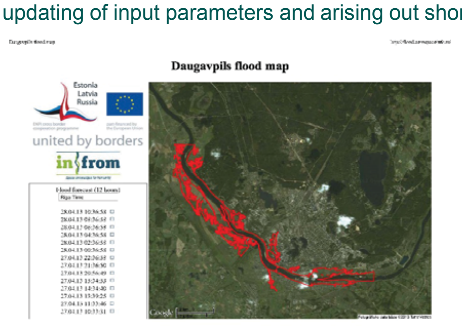
Based on the water flow discharge data and a digital elevation model, LISFLOOD-based hydrological model is built which allow forecasting inundation territories along the river basin.

To test and validate the model, the flood simulation results have been compared with available historical data on the flooded zones in the research area in March-April, 2010. The bounds of the inundation area from simulation experiments are close to historical data, and a forecast error is less that 10%

Performed real-time experiments with the developed model allowed achieving about 90% confidence in flood forecasts regarding significant objects which were actually inundated later on. The high forecast precision is achieved through continuous updating of input parameters and arising out short-term forecasts.



Matching between simulation results and historical data

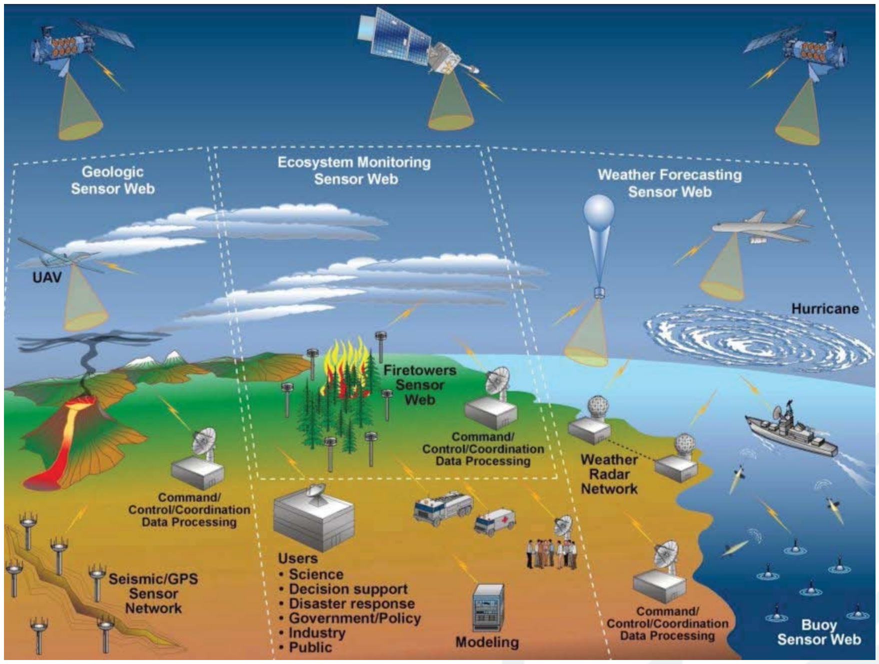


Sample flooding forecast in Daugavpils

The LISFLOOD-HP hydrological model generates 12-hour forecasts of inundation zones hourly. The results of flood simulation are presented as a raster map with information about the depth of water in the flooded territory which automatically vectorized to provide their compatibility with the external GIS software and store them in the database as archival information about flood dynamics.

16

INFROM: Sensor Web



17

Social and Sensor Data Fusion

- The two popular data types, social and sensor data, are, in fact, mutually compensatory in various data processing and analysis
- Participatory / citizen sensing, for instance, enables to collect people-sensed data via social network services over the areas where physical sensors are unavailable
- Simultaneously, sensor data is capable of offering precise context information, leading to effective analysis of social data
- Obviously, the potential of blending social and sensor data is high; nevertheless, they are typically processed separately, and the potential has not been investigated sufficiently

18

Nadezhda Kunicina,
professor, Dr.sc.ing., leading researcher
Faculty of Power and Electrical Engineering,
Institute of Industrial Electronics and Electrical Engineering,
1, Kalku Str., office 218, Riga, Latvia,
Contacts: +371-26162662
nadezda.kunicina@rtu.lv

Anatolijs Zabasta,
Dr.sc.ing., MBA, leading researcher
Faculty of Power and Electrical Engineering,
Institute of Industrial Electronics and Electrical Engineering,
12/1 Azenes Str., office 503, Riga, Latvia,
Contacts: +371-29232872
anatolijs.zabasta@rtu.lv

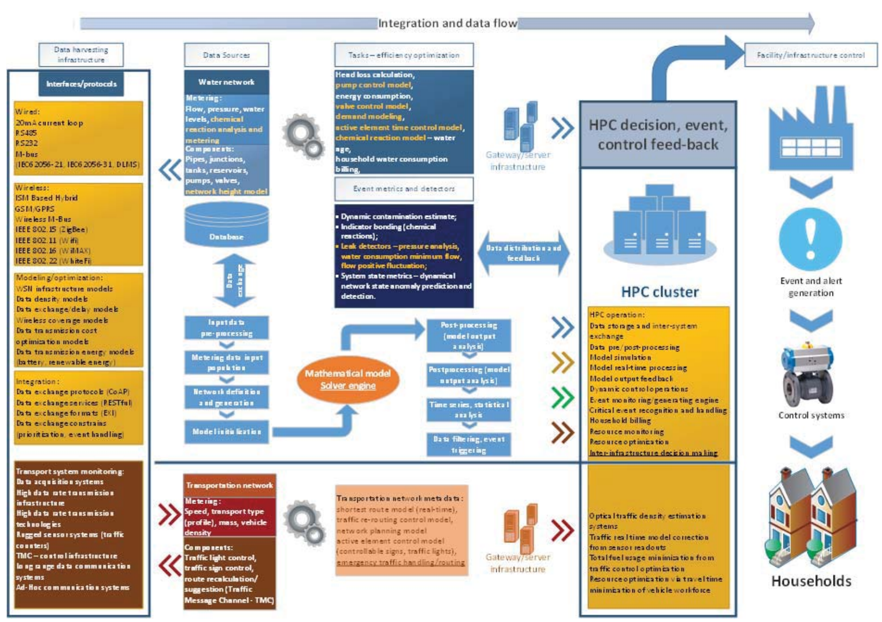
Applications domains

- The application domain includes overview of practical implementation of CPS, i.e. smart grid, autonomous automobile systems and process control systems.
- The application section gives an input in MPM4CPS COST action WG 3 objectives, in particular:
  - Collect the requests and requirements of each application domain, and rewrite them from a CPS perspective, look for commonalities/differences;
  - Assess the suitability of different application domain models from a CPS perspective (e.g., completeness, usability, interoperability with existing tools, etc.);
  - Compile recommendations on the proper use of different models and methodologies and the reliable assimilation of current application domain models in the perspective of CPS modeling.
- The experimental base of different infrastructure systems process control systems are leaded to creation of new System of System based control tool for city infrastructure services.

Abstract

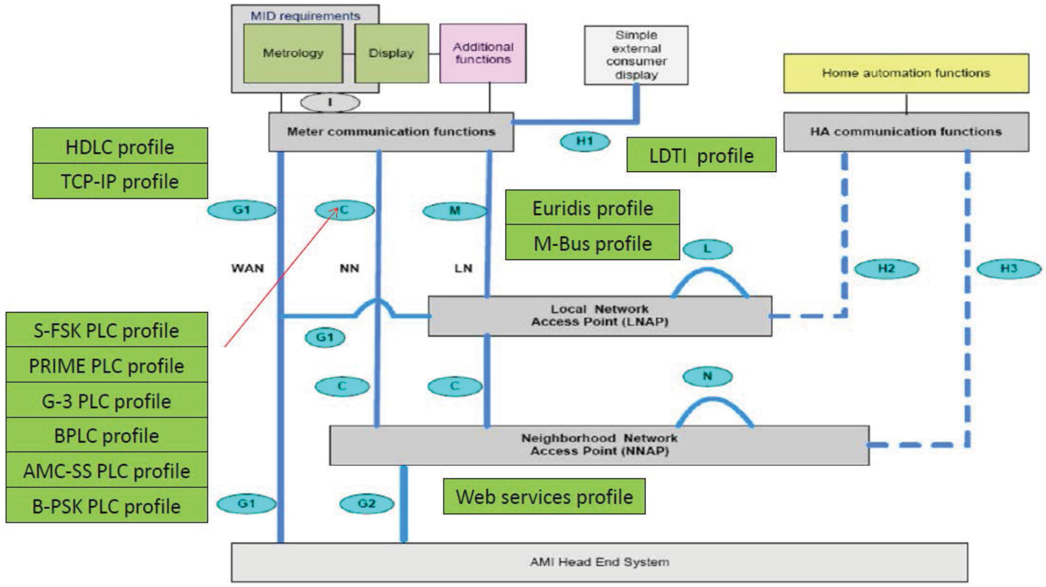
- The IoT supporting new generation of control concept is developed for smart city paradigm.
- The offered unified wireless data transmitting concept is deployed as city infrastructure and transport control approach.
- The developed solution is compatible with IoT on-line traffic management system and next generation of smart city control system.
- We have proposed a mapping of the IEC 62056 standard functionalities to a RESTful approach.
- Due to the paths resemble a fully-qualified file-name notation, they can be mapped to a URL, which makes the data model suitable for REST, therefore a REST URL could be written as: http://hostname/device/node/class/attribute.

21



22

Smart metering system reference architecture



23

The problem issues of utilities networks maintenance

- Each utility maintains its own network of meters, sensors and actuators, own system for data collection and storage, separate customer service, inventory, bookkeeping, billing and etc.
- The majority of the systems are obsolete and incompatible.
- Development of sensor network for monitoring and control and its maintenance is a challenging process due to plenty of human work.

24

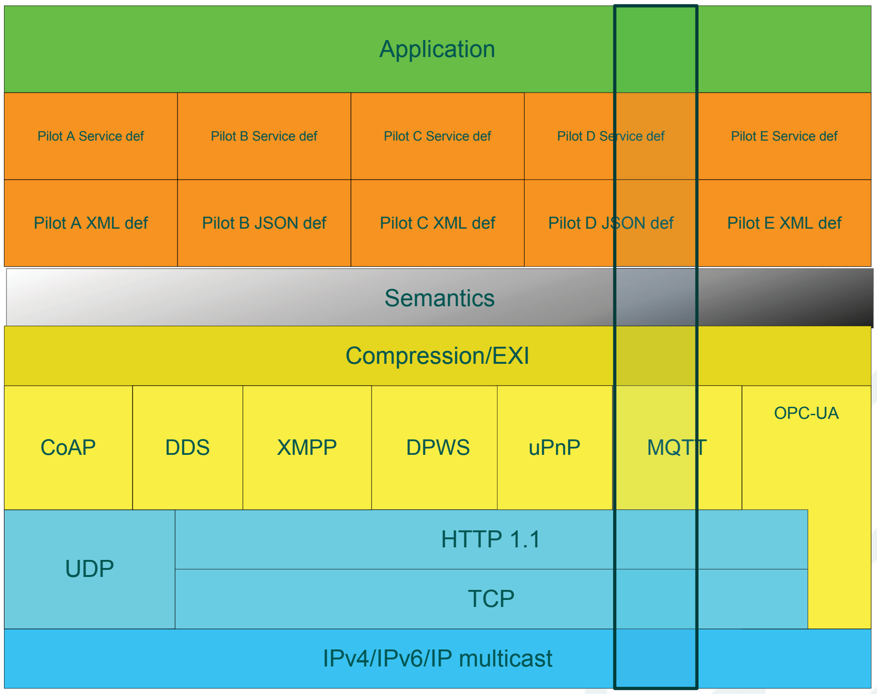
IEC/TS 62056 standards for automation in utilises

- "Electricity metering data exchange - The DLMS/COSEM suite".
- DLMS/COSEM protocol is not specific only to electricity metering, but it is also used for gas, water and heat metering:
  - IEC 62056-5-3:2013 DLMS/COSEM application layer
  - IEC 62056-6-1:2013 Object Identification System (OBIS) [17]
  - IEC 62056-6-2:2013 COSEM interface classes
  - ……………………….
  - The last one IEC/TS 62056-9-1 (in 2016) depicts use cases for DLMS with Web services and SOAP (Simple Object Access Protocol) interface. Messages are represented in XML format.

DLMS/COSEM vocabulary

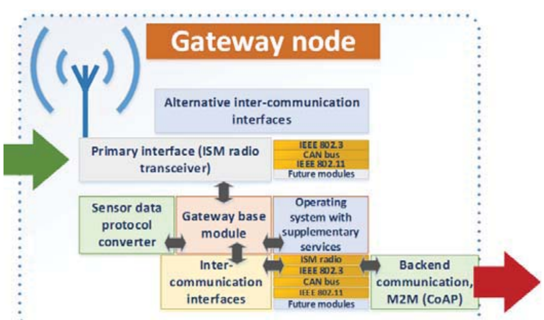
- COSEM “Companion Specification for Energy Metering”
  - Semantics: (meaning) of metering applications
  - object model to present metering functions through the interfaces
  - objects work together to realize the various functions
- OBIS “Object Identification System”
  - Naming system of the objects
- DLMS “Device Language Message Specification”
  - Syntax: application layer protocol specifying services to access COSEM objects
- DLMS/COSEM
  - Comprises the object model, the application layer protocol and the communication profiles to transport the messages

Service Oriented Protocols - a Challenge



27

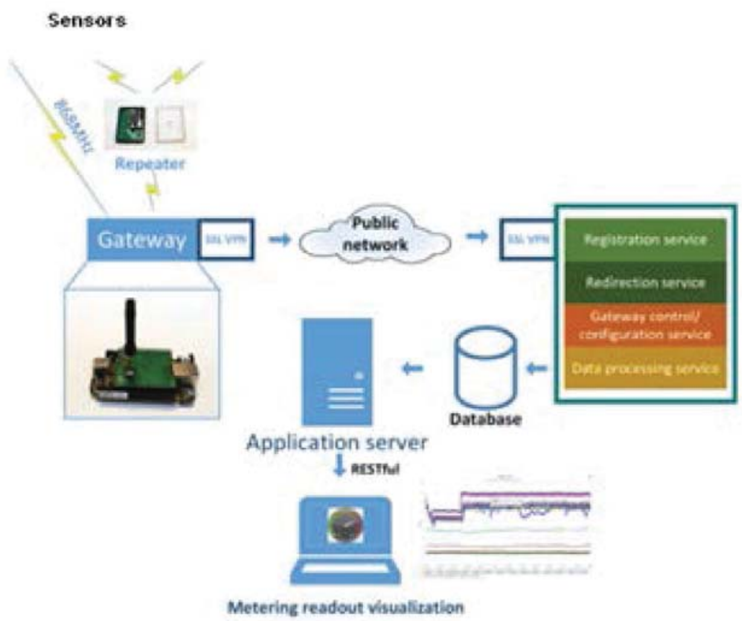
Hardware implementation



- The metering units consists of a microcontroller, readout interface, power supply and ISM band radio module operating at 868MHz. Transmission ranges of up to 500m are possible. The coded telegrams are transmitted using Manchester coding and GFSK modulation. To one or multiple gateway that provide area coverage.
- The gateway nodes consist or a radio module and 802.11b/g/n Wi-Fi module that integrates a microcontroller (ESP8266). The Wi-Fi module supports both station (Wi-Fi client mode) and access point modes. On start-up the gateway tries to connect to a Wi-Fi access point to access internet, if no access point is configured or not reachable, the gateway enter access point mode and starts broadcasting a SSID named after its MAC address. The owner of the gateway or the operator can use any PC or mobile equipment supporting 80.11b/g/n to connect to the gateway using a pre-shared password. The gateway automatically redirects the connecting client device to a configuration web interface.

28

Data transferring



29

An example of service humidity control

- For the ventilation and humidity control application two Core System services are used: CO_2 and Humidity by selected sensor location that is needed to be controlled. A third party control application that is not an internal development has been selected do demonstrates IoT system intercommunication using other independent service providers. For the purpose of demonstration power control hardware from ITEAD was used (https://www.itead.cc/smart-home.html). The power relay Sonoff is capable of handling loads up to 10A with a maximum rating of 2200Watts.
- This Wi-Fi enabled relay is able to communicate using MQTT or CoAP protocols with are of interest in the scope of the demo application.
- The 3rd part system is integrated into RTU demo system by creating a System of Systems interacting in a bidirectional way and performing smart monitoring and control applications.
- For the demo application two systems are controlled – a TwinFresh Comfo RA1-50 ventilator with regeneration system and a dehumidifier Ballu BDH-35L.

30

Conclusion
Potential feedback for MPM4CPS

- – Modelling and simulation of CP systems based on the two-hemisphere model-driven approach and other techniques and tools
- – Development of hardware for control of CP systems
- – Development of software for control of CP systems
- – Case studies and testing platforms

31