Concept of a Domain Repository for Industrial Automation

Camelia Maga and Nasser Jazdi

Institute of Industrial Automation and Software Engineering (IAS), Universität Stuttgart,
Pfaffenwaldring 47, 70569 Stuttgart, Germany
{Camelia.Maga, Nasser.Jazdi}@ias.uni-stuttgart.de

Abstract. Reuse approaches like domain engineering are increasingly shifted from software to system engineering. This results in new challenges to be faced by domain repositories, in which the reusable artifacts are stored. This paper describes the specific requirements for domain repositories for industrial automation systems and proposes a possibility to structure domain repository contents. The results of this paper are expected to be particularly interesting for researchers and practitioners in the area of industrial automation systems, since the proposed concept can be used in diverse industrial automation domains.

Keywords: Domain Engineering, Domain Repository, Industrial Automation, Reusable Artifacts

1 Introduction

Industrial automation spans a broad field of applications from product automation to industrial plants. For all these applications, there are numerous challenges to be faced like reduced time-to-market, reduced costs, increased variability and expectations concerning higher quality.

Domain engineering has been developed for software and offers a good approach for meeting these requirements, since it is based on reusability. Unfortunately, the adoption of this approach to industrial automation systems is not possible without major changes. Industrial automation possesses distinguishing characteristics, which require deeper research and new methodologies, in order to enable a systematic reuse.

A new approach, based on the domain engineering approach applied for software, is being currently developed at the Institute of Industrial Automation and Software Engineering of the Universität Stuttgart. The new approach considers the characteristics of industrial automation by taking not only software into account, but also hardware and the knowledge necessary to develop new industrial automation systems. The approach addresses three areas: domain engineering, application engineering and domain repository.

- Domain engineering is the process of analysis, design and realization of reusable artifacts within an industrial domain. It takes place decoupled of a certain customer order, in the form of preliminary work provided by domain-experts. The research activity on this area includes the development of a new methodology to identify, model, realize and test reusable artifacts in the industrial automation field. The proposed approach accounts for the industrial automation field by providing support to deal with the different disciplines involved, the sequence of steps required to create reusable artifacts and the needed auxiliary materials.

- Application engineering is the process that allows the creation of customer-specific industrial automation systems within a concrete customer order. The reusable artifacts created during domain engineering are deployed within the individual projects. The research activity on this area includes the development of a new methodology for application engineering, which is harmonized with domain engineering for industrial automation. It deals with the creation of new industrial automation systems from existing reusable artifacts, under consideration of the special needs of industrial automation.

- A domain repository is the storage area where the reusable artifacts are retained. It acts as an interface between domain engineering and application engineering, since it is created and gradually updated by the former process and deployed in the latter. The research activity on this area deals with the creation of a new domain repository for industrial automation, including the determination and the description of the reusable saved artifacts, the relations between them and the communication with domain engineering and application engineering.

The relationships among these areas are presented in Fig.1.

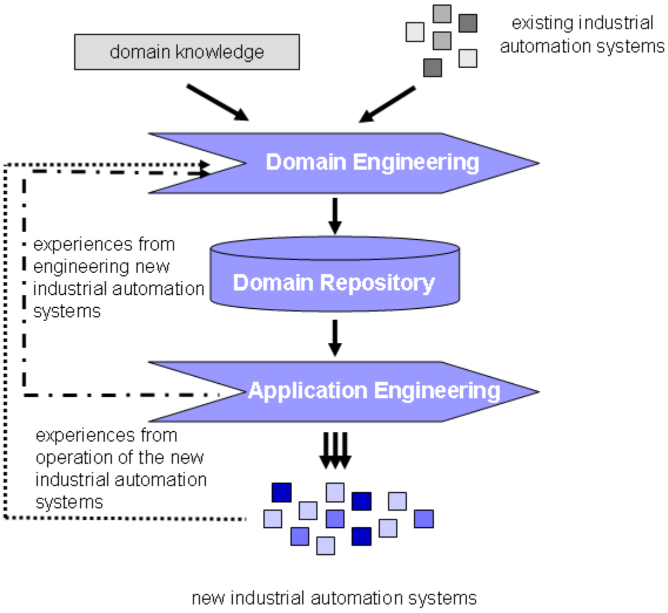
domain knowledge
existing industrial automation systems
Domain Engineering
experiences from engineering new industrial automation systems
Domain Repository
Application Engineering
experiences from operation of the new industrial automation systems
new industrial automation systems

Fig. 1: Overview of domain engineering, domain repository and application engineering.

The new approach with its constituents (i.e., domain engineering, application engineering, and domain repository) aims at satisfying the special requirements of industrial automation. The present paper focuses on the domain repository for industrial automation.

The paper is structured as follows: section 2 discusses the special requirements of industrial automation, which affect the domain repository. Section 3 proposes and illustrates the concept for a domain repository that fulfills the presented requirements. Section 4 concludes with a summary and provides an outlook for future work.

2 Requirements for an industrial automation domain repository

The concept of retaining reusable artifacts in a domain repository is not new [1]. The remarkable attention received by software reuse in academia has motivated researchers from different organizations to save, structure and retrieve reusable software components in many effective ways, which has brought about concepts for software component libraries, frameworks or software reuse repositories [2], [3], [4]. The migration of reuse concepts from software engineering to system engineering has not only brought new possibilities, but also new requirements for a domain repository.

The requirements for a domain repository to be used in industrial automation stem from two different directions: the construction of the domain repository (during domain engineering) and the deployment of the domain repository (during application engineering). The construction of the domain repository calls for an easy integration of new artifacts and a simple modification or deletion of the existing artifacts. The deployment of the domain repository requires easy retrieval of reusable artifacts and a clear, prescriptive description of their tailoring, in order to be used in the individual projects. These requirements are the same for classical software reuse repositories. Approaches for satisfying these requirements, like enumerated classification, faceted classification, free-text indexing, relational databases, and formal specifications, can be found in [2] and [5]. In this paper, the focus is on the distinguishing requirements for industrial automation.

Industrial automation systems consist of hardware (e.g. microcontroller, field bus, sensors and actuators) and software (e.g. control software, visualization software, communication software). Hence, a domain repository with reusable artifacts for industrial automation shall contain both hardware and software. This implies the requirement to consider the interdependencies between software and hardware and between different hardware artifacts. The challenge posed here is not just in the extra storage space required for hardware artifacts, but also in the different ways of hardware interaction that shall now be considered [6]. A hardware artifact has a physical form with well-defined dimensions and requires wiring, voltage supply and a physical location to be built at. It can be influenced by thermal or electromagnetic radiation from other hardware artifacts, so that all the information about compatibilities, recommendations and possible incompatibilities concerning their operation shall be saved together with the hardware artifacts in the domain repository.

The second specific requirement for domain repositories in industrial automation is the multitude of disciplines involved. Classical domain repositories contain only software components, so that only the expertise of software engineers is required. In addition to software engineers, a domain repository for industrial automation concerns automation engineers, mechanical engineers, electrical engineers, safety engineers or chemical engineers. These different disciplines have their own views, requirements and expectations on a domain repository. They aim at constructing and deploying the domain repository without directly taking into consideration the repercussions upon other disciplines. This raises the question of how the multitude of disciplines involved can be managed.

The third requirement on a domain repository for industrial automation considers the development processes, which are necessary to construct a new automation system. Since these development processes are almost constant within a certain domain (e.g. similar workflows, responsibilities, work results), the domain repository should regard them as reusable artifacts. This means that both development processes and their relations to the other reusable artifacts should be modeled in the domain repository.

Furthermore, systems in industrial automation are constructed with the help of auxiliary materials. These can be engineering or simulation tools, test benches, templates for different documents or special machines and assembly tools. Hence, a further requirement is to include the auxiliary materials in the domain repository and to connect them to the reusable artifacts, in order to optimize the application engineering process.

To summarize, beyond the typical requirements for a domain repository, there are specific requirements resulting from the intended application in industrial automation: the consideration of both hardware and software artifacts, the handling of numerous disciplines and the inclusion of development processes and auxiliary materials in the set of reusable artifacts. In the following section, a concept of a domain repository to satisfy these requirements is presented.

3 Structure of a domain repository for industrial automation

The classical requirements concerning the communication with the domain repository – during both its creation and its deployment – can be satisfied by providing a controlled access to the contents. This is enabled by the Domain Repository Engine (DRE), whose role is to manage the interaction with domain engineering and application engineering. In addition, it facilitates the internal interactions among the different partitions of the domain repository. It makes it possible to insert, delete, identify, extract, modify, search and manage changes in the domain repository contents. Possible principles for realizing the DRE are agent-based [7], [8], ontology-based [9], [10] as a workbench [2], or even as separate tool chain [11] combining the single functionalities listed above. Since the objective of the paper is to present the elementary structure of a domain repository for industrial automation, the realization of the DRE is not in the scope of this paper and is left to the choice of practitioners.

Specific requirements for industrial automation necessitate a suitable structure of the domain repository. Because hardware is modeled in such a repository, numerous new relations become possible. Compared with software components, which could exchange only signals, the components of an automation system can interact via signals, materials or energy. In addition, relations concerning incompatibilities, recommendations or alternatives between the single parts must be considered, as well. Due to the large number of possible relations and their diversity, we propose handling them in a separate layer, called the Connection Layer. The Connection Layer depicted in Fig. 2 is responsible for linking the different reusable artifacts. Based on [8] and [12], it is assumed that any new reusable artifact which is included into a domain repository already contains information regarding its internal structure, the existing and the required ports and the types of interconnections with other artifacts. In this way, the Connection Layer is able to use pre-defined style rules and to realize the linkage of the new artifact with the already existing ones. The idea is to execute these associations automatically, in order to avoid difficult and conflicting interconnections in case of insertion, modification or deletion. Another reason for introducing the Connection Layer is the existence of constraints, which affect more than one reusable artifact. Examples of constraints are the maximum accepted size of the entire industrial automation system, costs, performance, and maintainability aspects. These crosscutting constraints need to be supervised from a superior position in the domain repository. The artifacts to be used during application engineering are situated in the underlying layer, called the Basic Layer.

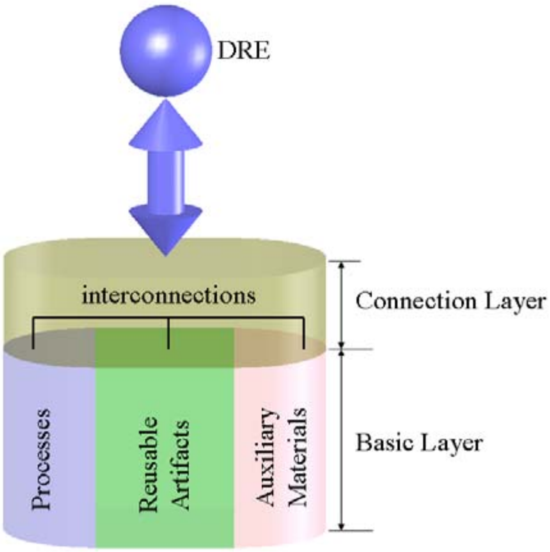

Fig. 2: Structure of the domain repository.

The members of each partition possess knowledge about their internal structure, the provided and the required ports and the types of interconnections with other artifacts from the same or a different partition. These interconnections within and between the partitions are managed by the Connection Layer. In the Processes partition, the workflows which are necessary to engineer a new industrial automation system are included. For instance, this partition contains in the case of industrial plants the engineering phases described in [13], and in the case of products the development phases described in [14]. The partition of Reusable Artifacts contains all elements that serve in constructing new industrial automation systems. Examples for such reusable artifacts are functional diagrams, piping layout models, installation and assembly drawings or process control system (PCS) instrumentation specifications. The Auxiliary Materials partition includes all artifacts that support the engineering of a new automation system, although not directly seen in the end result. Examples for Auxiliary Materials include software tools, assembly tools or special machines required during construction.

Through the integration of engineering processes and auxiliary materials in the domain repository and their linkage with the reusable artifacts over the Connection Layer, the domain repository provides real support of the project execution. Since the partitions are linked, it is possible to extract integrated results from the domain repository. For instance, consider a functional diagram, which is to be configured with the software tool x in the detailed engineering. During the configuration, possible contradictions with previously specified requirements can be detected. Similarly, incompatibilities or recommended functions for the current functional diagram are reported.

The description of the domain repository contents, within both Connection and Basic Layer, can be realized with non-formalized diagrams depicting graphically the reusable artifacts, the development processes, the auxiliary materials and their interconnections. As reported in [12], this representation may be ambiguous and cannot be analyzed in a formal way. These problems are tackled in software engineering by using Architecture Description Languages (ADLs). Readers interested in detailed discussion about ADLs are referred to [15]. For industrial automation, we propose to use an extended ADL, so that hardware, auxiliary materials, engineering processes and their interconnections can be modeled as well. In this way, we capitalize on the main strengths of an ADL, such as the support of alternative textual and graphical visualizations, the formalized interpretation and the ability to model internal structures, provided and required ports, types of interconnections and constraints [15].

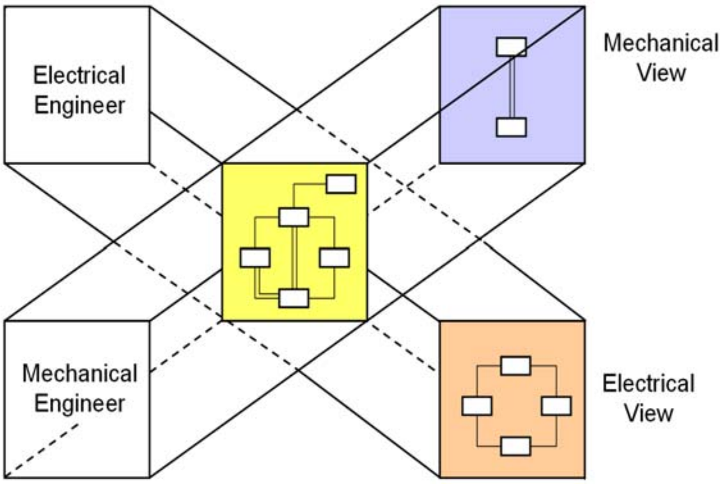

Fig. 3: Different views upon reusable artifacts of the domain repository.

In both creation and deployment of the reusable artifacts of the domain repository, there are many different disciplines involved. A solution to manage their multitude and interactions is to use discipline-specific views for the contents of the domain repository. Readers interested in detailed discussion about possible realizations are referred to [16] and [17]. According to [17], a view is a representation of a set of reusable artifacts from the perspective of a related set of concerns. For example, electrical and mechanical engineers perceive different configurations of the same industrial automation system, with discipline-specific properties inside the artifacts and discipline-specific interconnections between them, as depicted in Fig. 3. The concept of a discipline-specific view has an impact on the internal structure of the reusable artifact (e.g. for modeling discipline-specific properties), on the provided and the required ports, (like in case of modeling the wiring of a certain artifact), on the types of interconnections (e.g. for modeling signal, material or energy exchange) and on the constraints to be fulfilled (e.g. for modeling discipline-specific supervisions).

The principle of discipline-specific views applies for all contents of the domain repository, including discipline-specific development processes, reusable artifacts, and discipline-specific auxiliary materials. The Connection Layer realizes all interconnections within and between the single partitions of the domain repository. Discipline-specific views of the domain repository enable different visualizations of the contents. In other words, it enables focusing on development processes, reusable artifacts, auxiliary materials and interconnections for the concerned discipline, while hiding unnecessary domain repository contents.

4 Conclusions

This paper presents a concept of a domain repository for industrial automation. After discussing the specific requirements for an industrial automation domain repository, a two-layered structure of the domain repository is proposed. This contains reusable hardware and software artifacts, as well as development processes and auxiliary materials, which are necessary to develop new industrial automation systems. The characteristics of the presented concept of a domain repository are the deployment of entire industrial automation systems, the integration of development processes and auxiliary materials in the domain repository, the linkage between all reusable artifacts and the discipline-specific views upon the contents.

As future work, a further elaboration of the structure of the domain repository and evaluation based on a case study is planned. For this purpose, the complete domain engineering should be applied for a concrete domain. Afterwards the domain repository has to be developed and the application engineering process should be executed for the sake of evaluating the proposed approach.

References

1. Ebert, C., Smouts, M.: Tricks and Traps of Initiating a Product Line Concept in Existing Products. The 25th International Conference on Software Engineering (ICSE’03), Portland (2003)
2. Atkinson, C. et al: Handbuch zur komponentenbasierten Softwareentwicklung. Fraunhofer IESE and FZI, (2003)
3. Czarnecki, K., Eisenecker, U.: Generative Programming. Addison-Wesley Verlag, Boston, San Francisco, New York (2000)
4. Pohl, K., Böckle, G., Van der Linden, F.: Software Product Line Engineering: Foundations, Principles and Techniques. Springer Verlag (2005)
5. Henninger, S.: An Evolutionary Approach to Constructing Effective Software Reuse Repositories. ACM Transactions on Software Engineering and Methodology, Vol. 6, No. 2, 111--140 (1997)
6. Pahl, G., Beitz, W.: Konstruktionslehre Grundlagen. 7. Aufl., Springer-Verlag (2006)

7. Silverman, B., Bedewi, N., Morales, A.: Intelligent Agents in Software Reuse Repositories. CIKM Workshop on Intelligent Information Agents, Baltimore (1995)
8. Wagner, T.: Applying Agents for Engineering of Industrial Automation Systems. German Conference on Multiagent System Technologies (MATES), Erfurt (2003)
9. Billig, A., Sandkuhl, K.: ODIS – Ontology-based Domain Repository. 2nd Ljungby Workshop on Information Logistics, Ljungby (2004)
10. Braga, R., Werner, C., Mattoso, M.: Using Ontologies for Domain Information Retrieval. Proceedings 11th International Workshop on Database and Expert Systems Applications (DEXA 2000), IEEE Computer Society, Los Alamitos, California (2000)
11. Henninger, S.: Supporting the Domain Lifecycle. Proceedings of the International Workshop on Computer Aided Software Engineering, 10--19, Toronto, Canada (1995)
12. Medvidovic, N., Taylor, R.N., Whitehead, E.J.: Formal Modeling of Software Architectures at Multiple Levels of Abstraction. Proceedings of the California Software Symposium, 28--40, Los Angeles (1996)
13. DIN Deutsches Institut für Normung e.V.: DIN 28000-1 Chemical apparatus – Types of documents in the life cycle of process plants – Part 1: Registration of the essential and supplementary types of documents. Beuth Verlag, Berlin (2002)
14. DIN Deutsches Institut für Normung e.V.: DIN ISO 15226 Technical product documentation – Lyfe cycle model and allocation of documents. Beuth Verlag, Berlin (1999)
15. Medvidovic, N., Taylor, R.N.: A Classification and Comparison Framework for Software Architecture Description Languages. IEEE Transactions on Software Engineering, Volume 26, Issue 1, 70—93, IEEE Press Piscataway, New Jersey (2000)
16. Dijkman, R., Quartel, D., Pires, L., van Sinderen, M.: An Approach to Relate Viewpoints and Modeling Languages. Proceedings of the Seventh IEEE International Enterprise Distributed Object Computing Conference (EDOC'03), Brisbane (2003)
17. IEEE Standard on Architectural Description 1471, http://www.enterprise-architecture.info/Images/Documents/IEEE%201471-2000.pdf