An analysis on Industrial automation: a model-driven approach to requirements engineering

Dr. Shamita Chakaborty
Director
M. M. College Of Technology, Raipur (C.G.)
director@mmctraipur.com

ABSTRACT

Model Driven Requirements Engineering (MDRE) is a proposal to deal with the increasing complexity of technological systems by presenting requirement specifications as formal models that are correct, complete, consistent, unambiguous, and easy to read and maintain. This paper mainly discusses about the analysis on Industrial automation: a model-driven approach to requirements engineering.

Keywords: Industrial Automation Systems, Model Driven Requirements Engineering

1. Introduction

Model-driven development (MDD) is an approach for developing systems and software that emphasizes the usage of models throughout the development process. Models in MDD adhere to modeling languages with formal metamodels, such as the unified modeling language (UML). Model transformations can be used to change current models and build new, refined models in addition to manual development labor. The usage of transforms can help to automate error-prone operations like importing data from previous development phases and tools into models. Design models can be used to generate code or to assess the systems that have been produced. Model checks that are automated may discover flaws and discrepancies in models and between phase products.

In the discipline of software engineering, model driven engineering (MDE) has been shown to be capable of dealing with complexity. The use of MDE concepts to technical systems consisting of hardware and software components is a growing trend in modeling and simulation. With the increasing complexity of technological systems, various issues have arisen, such as maintaining design consistency and approving correctness in relation to client needs.

MDD is a software development paradigm that allows users to create complicated applications by abstracting pre-built components into simpler abstractions. Rather than telling, these visual building blocks demonstrate the business requirement and solutions to technical issues. MDD is the most crucial principle of low-code development: it's the link that connects IT and business domain specialists, allowing them to collaborate and turn ideas into useful solutions.

MDD works behind the scenes to eliminate human-process interference and simplify complexity through abstraction. Instead of being interpreted into code, the model in model-driven development projects is executable at runtime. MDD is able to avoid frequent operational and quality difficulties with code-centric projects as a result of this. Mendix's MDD strategy is also open and flexible, ensuring that your company is ready to quickly adapt and embrace new technology.

According to studies conducted by the Bosch Group, almost half of all field problems are caused by insufficient requirements engineering (RE). The RE is present throughout the entire product development process, which involves multiple engineering specialties. As a result, a global and standardized modeling language is needed to ensure that engineers from various disciplines understand one other. This standard language will enable the creation of domain-specific requirements models, system design models, traceability models, and verification models.

In recent years, there has been a lot of research into the application of model-driven methodologies in the field of industrial control. Modeling of control application needs, architecture, and details has long been seen as a significant aspect of design processes and a means of dealing with the applications' growing size and complexity. Many modern approaches have incorporated simulations into development processes in order to test early and concurrently with development activity.

2. Literature review

D. Hastbacka, T. Vepsalainen, and S. Kuikka (2011), T. Ritala and S. Kuikka (2013), T. Vepsalainen and S. Kuikka (2013) (2007), The authors have previously built simulator connection to the Aukoton MDD process for automation and control applications using the tool-supported Aukoton MDD process. The strategy is founded on the UML Automation Profile (UML AP). It allows for the modeling and simulation of cyclically executed control functions like as feedback, binary control, and interlocks (interlocks are used in control systems to protect the controlled processes from causing harm to themselves or personnel, e.g., by forcing actuators to safe states based on measured states of the processes).

The management and coupling of cosimulations have recently been handled with the FMI standard, according to the MODELISAR Consortium (2010).

Model-based strategies were used by G. Hemingway, H. Neema, H. Nine, J. Sztipanovits, and G. Karsai (2012). However, simulation approaches may not be the area of expertise of control application developers. As a result, using a single simulation engine can be deemed more beneficial.

T. Lukman, G. Godena, J. Gray, M. Hericko, and S. Strmcnik (2013) introduce the MAGICS technique for MDD of industrial process control software in the industrial control domain. ProcGraph, which has been built on the Eclipse platform on top of Eclipse Modeling Framework, is used as a modeling notation in this technique (EMF). Entity Diagrams (ED), State Transition Diagrams (STD), and State Dependency Diagrams (SDD) are among the diagram types used in the technique, with STD being particularly useful for representing sequential action. The technique allows for the creation of executables but not simulations.

The International Electrotechnical Commission's (IEC) definition and modeling language for industrial control applications was published in 2012. It builds on another IEC programming language's function block paradigm.

With event-driven execution and support for application deployment, the International Electrotechnical Commission (2013). IEC 61499 models can also be utilized for simulation purposes with the right tool support.

A model-integrated design framework for creating and validating industrial automation systems was developed by V. Vyatkin, H. Hanisch, C. Pang, and C. Yang (2009). It uses the IEC 61499 architecture and is based on the Intelligent Mechatronic Component (IMC) idea. IMCs are integrated together and their models enable formal verification, closed-loop MiL simulation of IEC 61499 models, and code deployment in new systems.

The MEDEIA project's methodology is based on the employment of numerous model types as well as bidirectional model transformations, according to I. Hegny, M. Wenger, and A. Zoitl (2010). The procedure allows for closed-loop MiL simulations in an IEC 61499 environment. The technique defines simulation models of process sections using either timed state charts or external behavior descriptions (external simulation tools).

3. Requirements Engineering in the Field of Industrial Automation Systems

Industrial automation systems are defined by their ability to process a material or work piece according to a set of instructions in order to produce the intended product. Even though boundary variables such as climate, disturbances, and material qualities may vary within a given range, the product's quality must stay stable. The process is automated in order to satisfy the quality goals in a repeatable, efficient, and consistent manner.

The systems engineer must now construct a machine capable of running the procedure in a deterministic manner. This activity is usually completed within a certain design domain, which refers to a technical field of competence such as treatment technology, mechanical design, driving system, and control. The treatment is frequently regarded as the original equipment manufacturer's main competency (OEM). The same may be said about mechanical design, which is closely connected.

4. Requirements Verification

The purpose of requirements verification in the MDRE process, according to Hongchao, is to verify the design against the requirements in an automated and repeatable manner. This is accomplished in the MDRE4BR profile by combining the enhanced semantics of the verify relation mentioned above with the concept of violation monitors and variable binding outlined in the vVDR methodology.

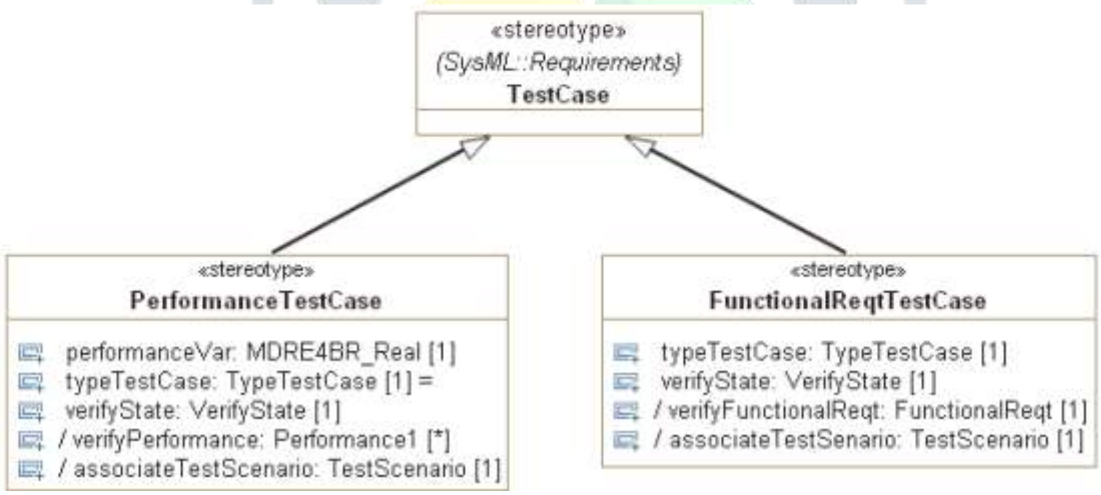

Fig 1: The performance and functional test cases [11]

Different sorts of test cases for various types of requirements are described as stereotypes in the MDRE4BR profile, such as the performance test case and the functional requirement test case, both of which are derived from the SysML meta class TestCase, as illustrated in Figure. In order to analyze the requirement, the violation monitor is modeled as part of the performance test case or functional test case.

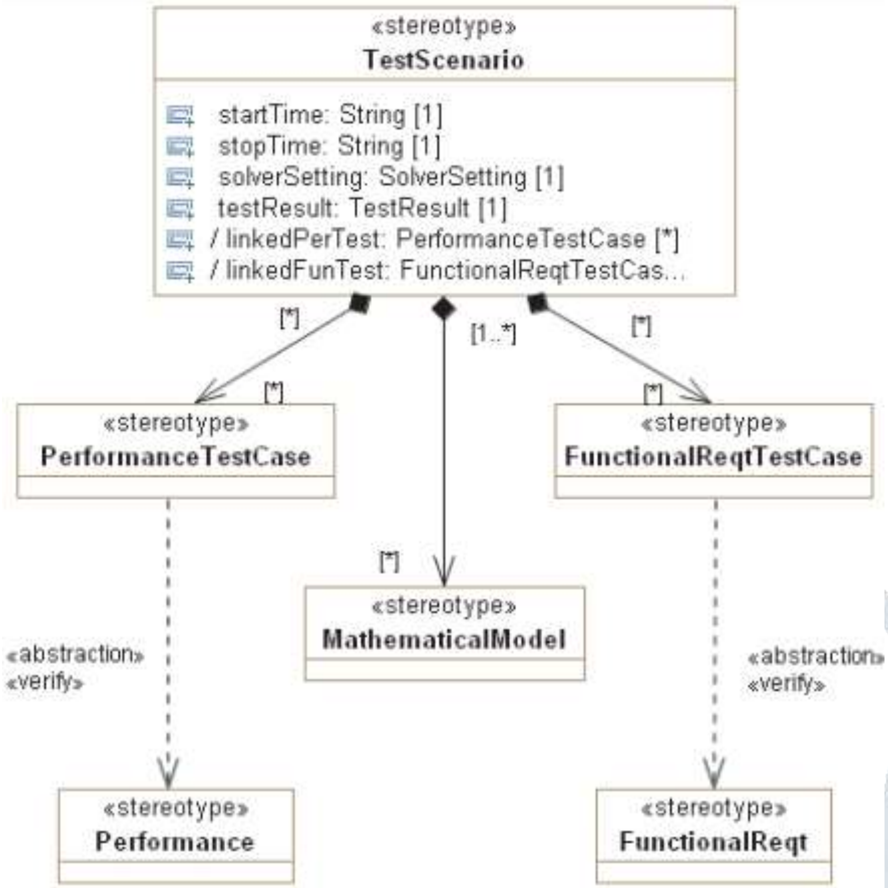

Figure 2: Verification Package Relationships [11]

Multiple test cases referring to at least one mathematical model are described as a new archetype called test scenario. Executing the test scenarios and related test cases described in the analytical model can be used to verify requirements. The relationships between these factors are represented in the diagram above.

Conclusion

Typically, providers provide the powertrain and control technology. This is because power supplies, actuators, and controllers are all accessible on the market as low-cost, high-quality standard components. Despite this, the drive and control system is inextricably linked to other components of the automation system. The components' right selection and integration into the overall structure have a significant impact on function, performance, robustness, and dependability. Interdisciplinary system design skill determines cutting-edge technologies.

References

1. T. Vepsalainen and S. Kuikka (2013), "Simulation-based development of safety related interlocks," in *Simulation and Modeling Methodologies, Technologies and Applications*, pp. 165–182, Springer

2. D. Hastbacka, T. Vepsalainen, and S. Kuikka (2011), "Model-driven development of industrial process control applications," *Journal of Systems and Software*, vol. 84, pp. 1100–1113

3. T. Ritala and S. Kuikka (2007), "UML automation profile: enhancing the efficiency of software development in the automation industry," in *Proceedings of the 5th IEEE International Conference on Industrial Informatics (INDIN '07)*, pp. 885–890

4. MODELISAR Consortium (2010), "Functional Mock-up Interface for Co-simulation," Version 1.0.

5. G. Hemingway, H. Neema, H. Nine, J. Sztipanovits, and G. Karsai (2012), "Rapid synthesis of high-level architecture-based heterogeneous simulation: a model-based integration approach," *Simulation*, vol. 88, no. 2, pp. 217–232

6. T. Lukman, G. Godena, J. Gray, M. Hericko, and S. Strmcnik(2013) , "Model-driven engineering of process control software-beyond device-centric abstractions," *Control Engineering Practice*, vol. 21, no. 8, pp. 1078–1096,.

7. International Electrotechnical Commission (2012), *IEC 61499-1: Function Blocks-Part 1: Architecture*, International Standard, Geneva, Switzerland, 1st edition

8. V. Vyatkin, H. Hanisch, C. Pang, and C. Yang (2009), "Closed-loop modeling in future automation system engineering and validation," *IEEE Transactions on Systems, Man and* Cybernetics Part C: Applications and Reviews, vol. 39, no. 1, pp. 17–28

9. International Electrotechnical Commission (2013), IEC 61131-3: Programmable Controllers part 3, Programming Languages, IEC Publication,.

10. I. Hegny, M. Wenger, and A. Zoitl (2010), "IEC 61499 based simulation framework for model-driven production systems development," in Proceedings of the 15th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA '10)

11. Hongchao Ji1 Oliver Lenord, Dieter Schramm (2011), A Model Driven Approach for Requirements Engineering of Industrial Automation Systems, 4th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools. ETH Zürich, Switzerland