The Role of Metamodeling in Systems Development

T

Balsam A. J. Mustafa
https://orcid.org/0000-0002-2973-7124
Al Hadi University College, Iraq

Mazlina Abdul Majid
University Malaysia Pahang, Malaysia

INTRODUCTION

What is Meta-Model? The word *"Meta"* literally means "after", "beyond" in Greek or more "comprehensive" (*Merriam-Webster's*, 1993). In computer science, it is widely used in different meanings. In Database Management Systems, metadata means (data about data) which may represent data dictionaries, repositories, etc., and models represent data like the ER model (Entity-Relationship). In other words, a meta-model is a model of the data model. In Programming Languages, a meta interpreter is an interpreter of a (program) interpreter (Smith, 1984). Meta-modeling is a method for defining the abstract syntax of a language, both modeling or programming language. It makes the development of a language simpler allowing the designers to directly map the classes identified in domain analysis to classes in the meta-model (Kleppe, 2008). The meta-model expresses what models include such as concepts and relationships between them and may be the rules of how these concepts can be interrelated. Hence, a metamodel can be treated just like any conceptual model of information systems. The only specialty is that the artifact of meta-modeling is a model (Jeusfeld, 2009), i.e. a model is an instance of a metamodeling, for example, any UML class diagram can be seen as an instance of the UML metamodel that should be well formed with respect to it (Osis & Donins, 2017).

At the beginning of the development of any system, the representation of a system view takes place once it is represented by a model. Kuhne (2006). defined a model as an abstraction of a real or language-based system allowing predictions or inferences to be made by developers. Kuhne (2006). explained that any model is built according to a specific meta-model which consists of a collection of functional or structural elements and rules to allow modeling the system view. The developer then can explain his ideas and discuss the conceptual view of the system with other stakeholders and can be further refined based on feedback from others. For any modeling method, its accuracy depends on the meta-model which semantically supports the features and behavior of the system that the method is used to model, i.e. the metamodel identifies the semantics of the system representation at the model level. Basha et al. (2012). pointed out that metamodeling is important because it provides a means for the machine to read, write, and understand models that were created and interpreted only by people. From this perspective, meta-modeling plays a key role in automating model based system development (MBSD). With models understandable to computers, tools can be built for model creation and code generation.

DOI: 10.4018/978-1-7998-9220-5.ch145

Harel and Rumpe (2004). explained that machines use machine-readable languages for communication just like people use natural languages to communicate between them. Both kinds of languages whether they are natural, artiðcial, or programming languages contain a large variety of meaningful elements. Therefore, a language consists of a syntactic notation (syntax), which is a set of legal elements, together with the meaning of those elements, which is expressed by relating the syntax to a semantic domain. A metamodel is a model of a language that developers use to design and implement a system and its structure consists of the essential elements of the language such as the language concepts, its graphical syntax and its semantics, i.e. what the models and programs written in the language mean and how they behave (Atkinson & Kuhne, 2003).

To improve system development practice, it is important to understand how languages can be managed to respond to the developers' demands. Language driven development is elaborated in this chapter. The right language improves the productivity of developers by increasing the value of primary software artifacts in terms of how much functionality they deliver at the development stages, and by reducing the rate at which primary software artifacts become obsolete (Atkinson & Kuhne, 2003).

In this chapter, we discuss first the role and importance of a language used in system development with its features, and introduces the steps of the meta-modeling process. In the sequel, we review the meta-modeling standards and the relationship between meta-modeling and model-driven architecture (MDA). The chapter also discusses the different categories of meta-models and finally presents the areas in meta-modeling where there is a demand for more research suggesting some future work.

BACKGROUND

The architecture of a system involves what elements make up the system and how they work together to provide the functionality of the system. The Model Driven Architecture (MDA) is an approach for software systems development initiated by the object management group (OMG) in 2001 (OMG, 2001). Unlike the other standards of the OMG, the MDA offers to use models instead of the traditional source code. It defines a specification that separates the system functionality from implementation that uses a specific technology platform. The architecture of a system is a specification of the parts and connectors of the system and the rules for the interactions of the parts using the connectors (Shaw, & Garlan, 1996). The standard of MDA released by OMG contains a set of guidelines for structuring the specification as models. MDA focused on creating and modeling the software products (Yousaf et al, 2019). The model here is an abstraction or a representation of a certain aspect of the system or a domain of the real-world that the system is designed for. The model aims to simplify the complexity of the system, focuses on the abstraction away from code to form a graphical model, which enables developers to understand, communicate, design, and implement the systems or adapt existing models. This makes the development of an application easier for those without prior coding knowledge. Different standard notations are used in modeling, e.g. the unified modeling language (UML).

MDA hence relies on models to be the main artifact in the development process to raise the level of abstraction to manage the complexity and change of the development process. This includes all types of models defined in the OMG standard which are the platform-independent model (PIM) and the platform-dependent model (PDM) to cover all system aspects in the development lifecycle. Models are created by a language, and the OMG' Meta-Object Facility (MOF) is introduced and clearly stated as the language in which all the languages for MDA are written (MOF, 2002). Modeling languages are used to define models, thus, their syntax and semantics (meaning) must be precisely defined. Hence, the process of