A Generic Method for Defining Viewpoints in SysML

Takahiro Yamada
Japan Aerospace Exploration Agency/Institute for Space and Astronautical Science
3-1-1 Yoshinodai, Sagamihara 229-8510, JAPAN

Abstract. It is important for system engineers to present the information on the systems that they are developing in a precise way so that the information can be understood by as many people as possible. The OMG Systems Modeling Language (SysML) (OMG 2008) was developed to meet that kind of needs of system engineers. It is a common practice to present descriptions of large systems using multiple views, each of which addresses one or more of the concerns of the system stakeholders. SysML has the concepts of view and viewpoint and it requires that a view conform to a viewpoint, which specifies conventions and rules for constructing and using the view, but it does not specify any method for specifying viewpoints.

One of the examples of specifying viewpoints in a particular problem domain is an ISO/IEC standard called the Reference Model of Open Distributed Processing (RM-ODP). This standard specifies five viewpoints to describe distributed processing systems, but it does not provide generic rules on how to construct viewpoints.

This paper proposes a generic method for defining viewpoints in SysML by generalizing the concepts used in defining viewpoints in RM-ODP. By using this method, viewpoints to be used in different problem domains can be defined in a systematic way, and reuse and/or sharing of viewpoints across different problem domains can also be facilitated.

Introduction

It is very important for system engineers to present the information on the systems that they are developing in a precise way so that the information can be understood by as many people as possible. The OMG Systems Modeling Language (SysML) (OMG 2008) was developed to meet that kind of needs of system engineers. SysML is a general-purpose modeling language for systems engineering applications and supports the specification, analysis, design, and verification and validation of a broad range of complex systems, including hardware, software, information, processes, personnel, and facilities.

It is a common practice to present descriptions of large systems using multiple views (for example, functional view, physical view, technical view, etc). Each of these views addresses one or more of the concerns of the system stakeholders. IEEE Recommended Practice for Architectural Description of Software-Intensive Systems (IEEE 2000) defines the concepts of viewpoint and view to set rules on what views are and how views should be constructed. In particular, a viewpoint is a specification of rules for constructing a view to address a set of stakeholder concerns, and the view is intended to represent the system from this viewpoint. This enables system engineers to describe a complex system with a set of views, each of which describes those aspects of the system specified by the corresponding viewpoint.

SysML has adopted the concepts of viewpoint and view defined in the IEEE recommended practice. SysML requires that a view conform to a viewpoint, which specifies conventions and rules for constructing and using the view, but it does not specify any method for specifying viewpoints.

One of the examples of specifying viewpoints for a particular problem domain is an ISO/IEC standard called the Reference Model of Open Distributed Processing (RM-ODP) (ISO/IEC 1996a and 1996b). This standard specifies five viewpoints (Enterprise, Information, Computational, Engineering, and Technology Viewpoints) to describe distributed processing systems.

If multiple views, each addressing a set of concerns, are used to describe a complex system, the job of describing the system becomes easier because the job of describing the complex system is decomposed into smaller jobs of describing sets of particular aspects of the system. For the same reason, the job of understanding a complex system becomes easier if multiple views are used to describe the system. Furthermore, if a set of standard viewpoints is defined to describe systems in a particular domain (for example, information systems, automobiles, spacecraft, etc.) and systems in that domain are always described with the views that conform to the standard viewpoints, it becomes still easier to describe and understand the descriptions because the definition of the viewpoints can be shared by different systems.

Neither the IEEE recommended practice nor RM-ODP provides generic rules on how to construct viewpoints. RM-ODP defines five viewpoints as explained above, but these viewpoints are specified independently. If there were generic rules on how to construct viewpoints, viewpoints could be defined in a more concise and harmonized way.

This paper proposes a generic method for defining viewpoints in SysML by generalizing the concepts used in defining viewpoints in RM-ODP. By using this method, viewpoints to be used in different problem domains can be defined in a systematic way, and reuse and/or sharing of viewpoints across different problem domains can also be facilitated.

Standard Viewpoints

RM-ODP (ISO/IEC 1996b) specifies five viewpoints (Enterprise, Information, Computational, Engineering, and Technology Viewpoints) to describe distributed processing systems. Having a standard set of viewpoints to describe systems in a particular domain helps both authors and readers of descriptions in various ways. If there are no standard viewpoint definitions, each author needs to define their viewpoints, and the readers need to understand the definitions of the viewpoints developed by individual authors. Since there is no guarantee that the viewpoints defined by some author are consistent with those defined by other authors, there can be cases in which similar systems are described differently using different viewpoints. This hinders understanding of descriptions of systems and sharing of information on systems. If there are standard viewpoints available for a particular domain, however, the authors do not to have to worry about defining viewpoints on their own, and the readers do not have to understand viewpoints for every new description.

The five viewpoints defined by RM-ODP were developed for describing distributed processing systems. The basic concepts of these viewpoints can be applied to other domains than distributed processing, but there are cases in which systems cannot be described well enough with the five viewpoints of RM-ODP. One such example is the domain of space data systems. The Consultative Committee for Space Data Systems (CCSDS) developed a reference architecture for describing space data systems called the Reference Architecture for Space Data Systems (RASDS) (CCSDS 2008) based on RM-ODP. RASDS defines five viewpoints (Enterprise, Information, Functional, Connectivity, and Communications Viewpoints). They are defined based on the RM-ODP viewpoints but there are some differences between the RASDS and RM-ODP viewpoints because the concerns of the stakeholders of space data systems are not the same as those of the stakeholders of distributed processing systems. For example, the connectivity of the physical elements in a space data system is a great concern to the stakeholders of space data systems. Therefore, a viewpoint to address the connectivity of physical elements has been defined in RASDS, which is called the Connectivity Viewpoint.

There may be other domains that will benefit from having a set of standard viewpoints for describing systems in those domains. Furthermore, if there are generic principles of defining viewpoints across many domains, it will facilitate definition of domain-specific viewpoints and reuse and/or sharing of viewpoints among different domains.

The following sections propose a generic method for defining domain-specific viewpoints in a systematic way in SysML, but this proposal is still at a very preliminary stage and the formal definition of the concepts that are presented in this paper is yet to be developed.

Generic Viewpoint

In order for different viewpoints in different domains to be developed coherently, a general model that specifies how viewpoints should be constructed is to be defined. This general model is called the generic viewpoint in this paper. Any viewpoint is to be defined by specializing the generic viewpoint. The relationship between the generic viewpoint and viewpoints and the relationship between a viewpoint and a view is explained in Figure 1.

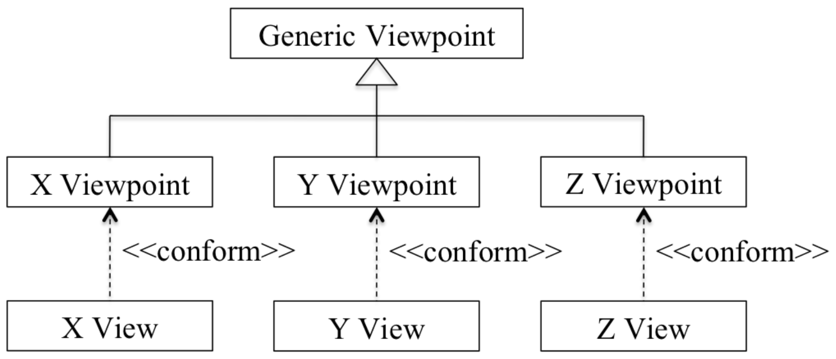

Figure 1. Relationship between the generic viewpoint and viewpoints and the relationship between a viewpoint and a view

The generic viewpoint proposed here was developed by generalizing the viewpoints defined in RM-ODP so that they can be used to define any viewpoint in any domain (not just the five viewpoints defined in RM-ODP) (Yamada 2007). The generic viewpoint is constructed with modeling elements used in UML and SysML.

The generic viewpoint specifies basic elements that can be used to define domain-specific elements in any domain-specific viewpoints, in which customized versions of these elements are used. These basic elements are presented in the next section.

Basic Elements

This section presents the definition of some of the basic elements of the generic viewpoint. Most of the terms defined here are based on those defined in UML (OMG 2007a and 2007b) and SysML (OMG 2008), but they are presented here as the basis for defining elements used to define specific viewpoints in specific domains.

The basic elements defined in this section can be used in any viewpoint where they are necessary. The same definitions of these elements must be used in any viewpoint, but different specializations of these elements can be specified in different viewpoints.

Block: (from SysML) A block is a modular unit that describes the structure of a system or element. It may include both structural and behavioral features, such as properties and operations that represent the state of the system and behavior that the system may exhibit. Some of these properties may hold parts of a system, which can also be described by blocks. A block may own ports (standard ports and/or flow ports).

Relationship: (from UML): Relationship is an abstract concept that specifies some kind of relationship between elements.

Property: (from UML) A property is a typed element that represents an attribute of a block.

Operation: (from UML) An operation is invoked in the context of a block.

Port: (from UML) A port is an interaction point between a block or part and its environment that is connected with other ports via connectors.

Standard Port: (from UML) A standard port represents an interaction point between a block and its environment.

Flow Port: (from SysML) A flow port is an interaction point through which input and/or output of items such as data, material, or energy may flow.

Connector: (from UML) A connector specifies a link that enables communication between two or more instances of blocks.

Item Flow: (from SysML) An item flow describes the flow of items across a connector.

Behavior: (from UML) For the purpose of this paper, the word behavior refers to any behavioral feature of the system being described.

Allocate: (from SysML) Allocate is a mechanism for associating elements of different types, or in different hierarchies, at an abstract level. In this paper, only behavior allocation (that is, allocation of behaviors to blocks) is considered for sake of brevity.

Figure 2 provides a UML-like conceptual model of some of the basic elements of the generic viewpoint and their inter-relationships. In the figure, boxes represent the basic elements mentioned in this section. Lines connecting boxes represent relationships between basic elements. A line may be labeled with a multiplicity. “1..*” is used to denote “one or more,” and “0..*” is used to denote “zero, one, or more”. A diamond at the end of a relationship line denotes a *part-of* relationship. For example, blocks are part of the generic view. A triangle at the end of a relationship line denotes a *kind-of* relationship. For example, standard ports are a kind of ports.

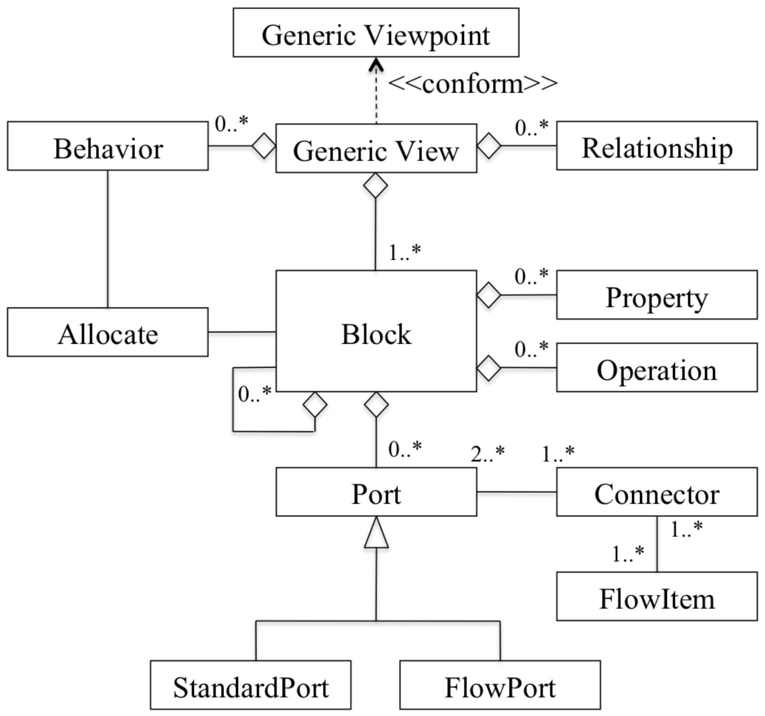

Figure 2. Conceptual model of the basic elements of the generic viewpoint

Defining Viewpoints

Specific viewpoints can be defined as SysML profiles by specializing the generic viewpoint. Specifically, a specific viewpoint can be defined by specifying the following items to be used in that viewpoint (not all of these items need to be specified for every viewpoint, though). These items are specializations of the basic elements defined in the previous section.

Specialized Block(s): Blocks used in the viewpoint that can be specified by specializing block. These may include blocks that are specified by further specializing specialized Blocks.

Specialized Propertie(s): Properties that represent attributes of the specialized blocks defined in the viewpoint.

Specialized Operation(s): Operations invoked in the context of the specialized blocks defined in the viewpoint.

Specialized Port(s): Ports representing interaction points between a specialized block defined in the viewpoint and its environment.

Specialized Connector(s): Links that enable communication between two or more instances of specialized blocks defined in the viewpoint.

Specialized Flow Item(s): Items that flow across a specialized connector defined in the viewpoint.

Specialized Relationship(s): Abstract concepts that specify some kinds of relationships between elements defined in the viewpoint.

Specialized Behavior(s): Behavioral features of the elements defined in the viewpoint.

Specialized Allocate(s): Mechanisms for associating elements defined in the viewpoint.

Viewpoints can also be defined by specializing a viewpoint that has already been defined. Such viewpoints are defined by further specializing elements defined in the original viewpoint and/or by adding more items to the original viewpoint.

Figure 3 provides a conceptual model of defining viewpoints based on the generic viewpoint. The graphical conventions used in this figure are the same as those used in Figure 2.

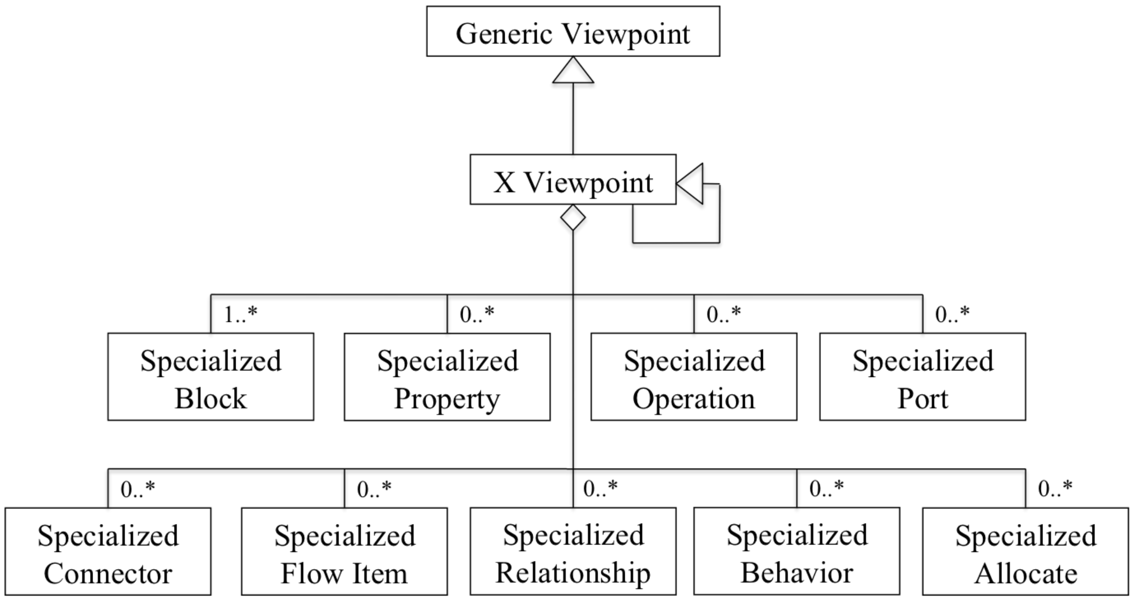

Figure 3. Conceptual model of defining viewpoints

Viewpoints that are defined for some problem domain and applicable to many systems in that domain should be published as a domain-specific standard so that they can be used universally in that domain. Viewpoints deemed useful for describing other systems should be kept in a viewpoint library so that they can be reused in the description of other systems either in their original forms or in specialized forms.

If a system is described using multiple viewpoints, these descriptions (that is, views of the system) must be mutually consistent. In order to assure that different views of a system are mutually consistent, it should be specified how elements in a viewpoint correspond to elements in other viewpoints. Correspondence between viewpoints can be specified in several ways. One way is to use the same instances of some block(s) in multiple viewpoints. In this case, these blocks have a different set of properties in each of the viewpoints in which they appear. Another way of specifying correspondence is to specify a relationship or an allocation between an element type in a viewpoint and another element type in another viewpoint.

An Example of Viewpoint Definition

This section provides an example of a viewpoint defined based on the generic viewpoint. The viewpoint presented here is a protocol viewpoint to be used to describe the structure of communications protocols used in distributed processing systems.

The protocol view of a system, which is described according to the protocol viewpoint, consists of protocols that interact with each other. A protocol is a specialized block defined for the protocol viewpoint and represents an element that executes a communications protocol. A protocol may have protocol properties, protocol operations, and protocol ports, etc. Each of these items is an element defined by specializing a basic element defined in the generic viewpoint.

Figure 4 provides a conceptual model of the protocol viewpoint defined in this section. The graphical conventions used in this figure are the same as those used in Figure 2.

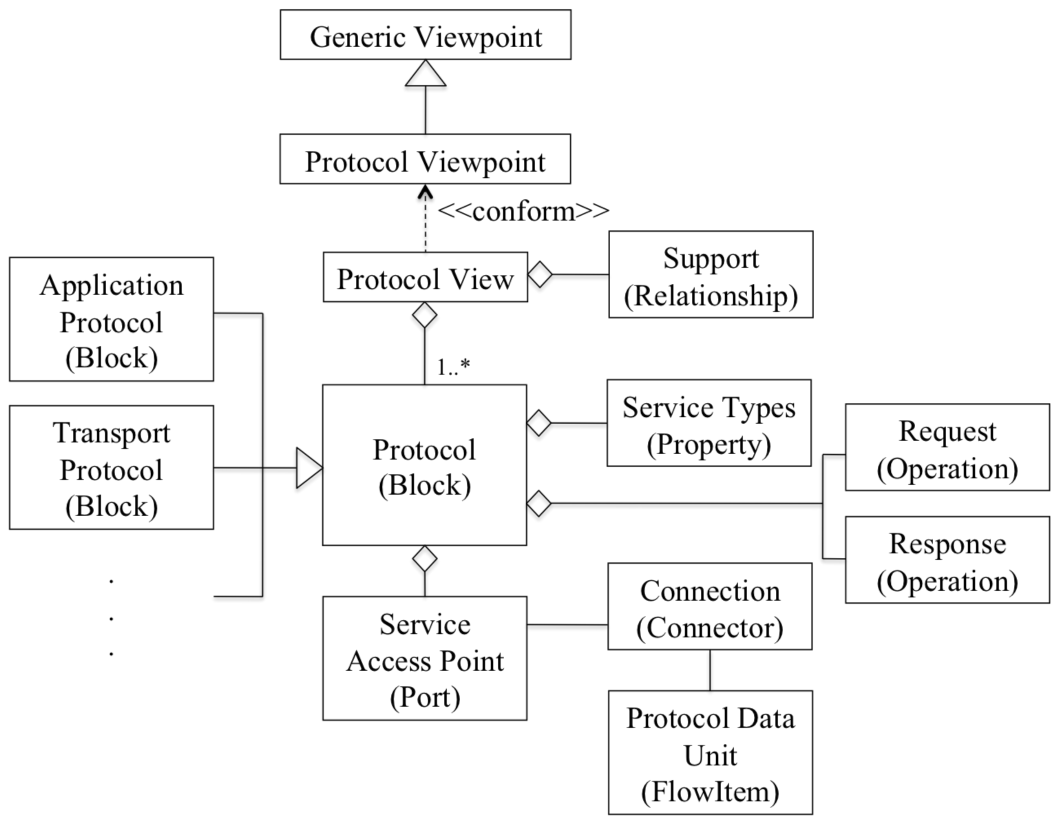

Figure 4. Conceptual model of a protocol viewpoint

Benefits of This Method

The method for defining viewpoints based on the generic viewpoint has several benefits.

SysML uses the concept of viewpoints defined by the IEEE Recommended Practice (IEEE 2000), but neither of them provides rules on how viewpoints should be constructed. Therefore, the users of SysML need to define viewpoints on their own. The method proposed here defines a generic viewpoint from which specific viewpoints are defined by specialization. In this way, different viewpoints can be defined in a systematic and coherent way.

Viewpoints defined for some problem domain should be published as a domain-specific standard so that they can be used universally in that domain. This facilitates exchange and/or sharing of information on different systems among projects in that domain. Further, viewpoints published as a domain-specific standard can be specialized further so that they can be applied to sub-domains of that domain.

Another benefit of this method is that it will facilitate drawing of domain-specific models (that is, views of a target system) with software tools because viewpoint definitions can be managed systematically. Software tools that support definition of new viewpoints based on the generic viewpoint and/or existing viewpoints can also be developed based on this method.

Conclusion

This paper proposed a generic method for defining viewpoints in SysML. This method uses a generic viewpoint that specifies how viewpoints should be constructed. Specific viewpoints can be developed by specializing the generic viewpoint. New viewpoints can also be developed by specializing existing viewpoints. In this way, different viewpoints can be defined systematically.

Viewpoints defined for some problem domain should be published as a domain-specific standard so that they can be used universally in that domain. This method will also facilitate drawing of domain-specific models (views of a target system) with software tools because viewpoint definitions can be managed systematically.

The proposal presented here is only preliminary and does not specify the generic viewpoint completely. It does not specify the method for defining specific viewpoints formally, either. This method should be refined further and published eventually as a standard of some kind so that specific viewpoints for any domain can be developed in a unified way.

The author hopes that this paper stimulates discussion on the development of such a generic viewpoint in the community of system engineering and in the communities that require an efficient way of presenting their systems.

References

Consultative Committee for Space Data Systems (CCSDS). 2008. Reference Architecture for Space Data Systems. Magenta Book. CCSDS 311.0-M-1.

Institute of Electrical and Electronics Engineers (IEEE). 2000. IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE Std 1471-2000.

ISO/IEC. 1996a. Information Technology - Open Distributed Processing - Reference Model: Foundations. ISO/IEC 10746-2.

ISO/IEC. 1996b. Information Technology - Open Distributed Processing - Reference Model: Architecture. ISO/IEC 10746-3.

Object Management Group (OMG). 2007a. Unified Modeling Language: Infrastructure. Version 2.1.1.

Object Management Group (OMG). 2007b. Unified Modeling Language: Superstructure. Version 2.1.1.

Object Management Group (OMG). 2008. OMG Systems Modeling Language. Version 1.1.

Yamada, T. 2007. Proposal for Defining a Generic Viewpoint in RM-ODP. 4th International Workshop on ODP for Enterprise Computing (WODPEC 2007).

BIOGRAPHY

Dr. Takahiro Yamada (tyamada@pub.isas.jaxa.jp) is the leader of the telecommunications and data handling group and the satellite operations group at the Institute of Space and Astronautical Science (ISAS) of Japan Aerospace Exploration Agency (JAXA), and is responsible for developing communications and data handling systems for space science projects of JAXA. He has developed communications and data handling systems for such projects as Hayabusa (asteroid sample return project), Suzaku (X-ray telescope project), and Hinode (solar telescope project). He is also actively involved in standardization of space data systems and has edited more than a dozen standards published by the Consultative Committee for Space Data Systems (CCSDS), including the Reference Architecture for Space Data Systems (RASDS). Presently, he chairs the Space Communications Cross Support Architecture Working Group of CCSDS.