Architecture Viewpoint Template for ISO/IEC/IEEE 42010

Rich Hilliard
r.hilliard@computer.org

VERSION 2.2

Abstract

This is a template for specifying architecture viewpoints in accordance with ISO/IEC/IEEE 42010:2011, *Systems and software engineering—Architecture description*.

Using the template

This is a template that architects and organizations can use for documenting an architecture viewpoint in accordance with ISO/IEC/IEEE 42010:2011 [6]. In particular, the requirements on viewpoints are found in Clause 7 of that Standard.

The template provides an outline for an architecture viewpoint and defines a set of "slots" or information items to be elaborated by the architect using the template to define and specify a viewpoint. Each slot is identified by a heading name followed by a brief description of its intended content and guidance for developing that content.

The template uses a few conventions, as follows.

⋆ "Musts" are items which must be present to satisfy the Standard. Musts are marked like this.

□ "Shoulds" are items recommended to be present, but not required by the Standard. Shoulds are marked like this.

Optional items are marked with this: (optional).

<Items> like <this> signal names to be filled-in by a user of the template and used throughout the viewpoint definition.

The source files for this version of the template is packaged as a LaTeX `section`. It is designed to be included in a LaTeX document such as those using the report, book or article document class. (Other versions for Microsoft Word and XML will also be made available.)

License

The *Architecture Viewpoint Template* is copyright © 2012–2014 by Rich Hilliard.

The latest version is always available at `http://www.iso-architecture.org/42010/templates/`

The template is licensed under a Creative Commons Attribution 3.0 Unported License. The terms of use are here:
`http://creativecommons.org/licenses/by/3.0/`

This license gives you the user the right to share and remix this work to define new architecture viewpoints. It does not require you to share the results of your usage (i.e., new viewpoint definitions), but if your use is non-proprietary, we encourage you to share your viewpoint definition with others for their use via the WG42 Viewpoint Repository
`http://www.iso-architecture.org/viewpoints/`.

Version History

This template is based on one originally designed for use with IEEE std 1471:2000 [5] which was published as [4]. That template formed the basis for the viewpoint template in [3], which appeared during the development ISO/IEC/IEEE 42010:2011 and subsequently appeared in Annex B of the published ISO/IEC/IEEE 42010:2011.

The present document is an enhanced version of these earlier templates, with additional guidance, clarifications and examples for readers.

rev 2.2 7 October 2014, Moved bibliography from `bibtex` to `biblatex`. Released revision with minor formatting fixes.

rev 2.1b June 2012, initial release on 42010 website.

Comments or Questions

Contact the author Rich Hilliard [r.hilliard@computer.org] for questions or comments. For more information on ISO/IEC/IEEE 42010, visit the ISO/IEC/IEEE 42010 website:
`http://www.iso-architecture.org/42010/`.

The template begins here . . .

1 <Viewpoint Name>

⋆ Provide the name for the viewpoint.

If there are any synonyms or other common names by which this viewpoint is known or used, record them here.

2 Overview

Provide an abstract or brief overview of the viewpoint.

Describe the viewpoint's key features.

3 Concerns and stakeholders

Architects looking for an architecture viewpoint suitable for their purposes often use the identified concerns and typical stakeholders to guide them in their search. Therefore it is important (and required by the Standard) to document the concerns and stakeholders for which a viewpoint is intended.

3.1 Concerns

⋆ Provide a listing of architecture-relevant concerns to be framed by this architecture viewpoint per ISO/IEC/IEEE 42010, 7a.

Describe each concern.

Concerns name “areas of interest” in a system.

NOTE: *Following ISO/IEC/IEEE 42010, **system** is a shorthand for any number of things including man-made systems, software products and services, and software-intensive systems such as “individual applications, systems in the traditional sense, subsystems, systems of systems, product lines, product families, whole enterprises, and other aggregations of interest”.*

Concerns may be very general (e.g., *Reliability*) or quite specific (*e.g., How does the system handle network latency?*).

Concerns identified in this section are critical information for an architect because they help her decide when this viewpoint will be useful.

When used in an architecture description, the viewpoint becomes a “contract” between the architect and stakeholders that these concerns will be addressed in the view resulting from this viewpoint.

It can be helpful to express concerns *in the form of questions* that views resulting from that viewpoint will be able to answer. E.g.,

- *How does the system manage faults?*
- *What services does the system provide?*

NOTE: *“In the form of a question” is inspired by the television quiz show, Jeopardy!*

ISO/IEC/IEEE 42010, 5.3 contains a candidate list of concerns that must be considered when producing an architecture description. These can be considered here for their relevance to the viewpoint being specified:

- What are the purpose(s) of the system-of-interest?
- What is the suitability of the architecture for achieving the system-of-interest’s purpose(s)?
- How feasible is it to construct and deploy the system-of-interest?
- What are the potential risks and impacts of the system-of-interest to its stakeholders throughout its life cycle?
- How is the system-of-interest to be maintained and evolved?

See also: ISO/IEC/IEEE 42010, 4.2.3.

3.2 Typical stakeholders

⋆ Provide a listing of the typical stakeholders of a system who are in the potential audience for views of this kind, per ISO/IEC/IEEE 42010, 7b.

Typical stakeholders would include those likely to read such views and/or those who need to use the results of this view for another task.

Stakeholders to consider include:

- users of a system;
- operators of a system;
- acquirers of a system;
- owners of a system;
- suppliers of a system;
- developers of a system;
- builders of a system;
- maintainers of a system.

3.3 “Anti-concerns” (optional)

It may be helpful to architects and stakeholders to document the kinds of issues for which this viewpoint is *not appropriate or not particularly useful*.

Identifying the “anti-concerns” of a given notation or approach may be a good antidote for certain overly used models and notations.

4 Model kinds+

⋆ Identify each model kind used in the viewpoint per ISO/IEC/IEEE 42010, 7c.

In the Standard, each architecture view consists of multiple architecture models. Each model is governed by a *model kind* which establishes the notations, conventions and rules for models of that type. See: ISO/IEC/IEEE 42010, 4.2.5, 5.5 and 5.6.

Repeat the next section for each model kind listed here the viewpoint being specified.

5 <Model Kind Name>

⋆ Identify the model kind.

5.1 <Model Kind Name> conventions

⋆ Describe the conventions for models of this kind.

Conventions include languages, notations, modeling techniques, analytical methods and other operations. These are key modeling resources that the model kind makes available to architects and determine the vocabularies for constructing models of the kind and therefore, how those models are interpreted and used.

It can be useful to separate these conventions into a *language part*: in terms of a metamodel or specification of notation to be used and a *process part*: to describe modeling techniques used to create the models and methods which can be used on the models that result. These include operations on models of the model kind.

The remainder of this section focuses on the language part. The next section focuses on the process part.

The Standard does not prescribe *how* modeling conventions are to be documented. The conventions could be defined:

I) by reference to an existing notation or language (such as SADT, UML or an architecture description language such as ArchiMate or SysML) or to an existing technique (such as $M/M/4$ queues);

II) by presenting a metamodel defining its core constructs;

III) via a template for users to fill in;

IV) by some combination of these methods or in some other manner.

Further guidance on methods I) through III) is provided below.

Sometimes conventions are applicable across more than one model kind – it is not necessary to provide a separate set of conventions, a metamodel, notations, or operations for each, when a single specification is adequate.

5.1.1 I) Model kind languages or notations (optional)

Identify or define the notation used in models of the kind.

Identify an existing notation or model language or define one that can be used for models of this model kind. Describe its syntax, semantics, tool support, as needed.

5.1.2 II) Model kind metamodel (optional)

A metamodel presents the AD elements that constitute the vocabulary of a model kind, and their rules of combination. There are different ways of representing metamodels (such as UML class diagrams, OWL, eCore). The metamodel should present:

entities What are the major sorts of conceptual elements that are present in models of this kind?

attributes What properties do entities possess in models of this kind?

relationships What relations are defined among entities in models of this kind?

constraints What constraints are there on entities, attributes and/or relationships and their combinations in models of this kind?

NOTE: *Metamodel constraints should not be confused with architecture constraints that apply to the subject being modeled, not the notations used.*

In the terms of the Standard, entities, attributes, relationships are *AD elements* per ISO/IEC/IEEE 42010, 3.4, 4.2.5 and 5.7.

In the *Views-and-Beyond* approach [1], each viewtype (which is similar to a viewpoint) is specified by a set of elements, properties, and relations (which correspond to entities, attributes and relationships here, respectively).

When a viewpoint specifies multiple model kinds it can be useful to specify a single viewpoint metamodel unifying the definition of the model kinds and the expression of correspondence rules. When defining an architecture framework, it may be helpful to use a single metamodel to express multiple, related viewpoints and model kinds.

5.1.3 III) Model kind templates (optional)

Provide a template or form specifying the format and/or content of models of this model kind.

5.2 <Model Kind Name> operations (optional)

Specify operations defined on models of this kind.

See §6 for further guidance.

5.3 <Model Kind Name> correspondence rules

⋆ Document any correspondence rules associated with the model kind.

See §7 for further guidance.

6 Operations on views

Operations define the methods to be applied to views and their models. Types of operations include:

construction methods are the means by which views are constructed under this viewpoint. These operations could be in the form of process guidance (how to start, what to do next); or work product guidance (templates for views of this type). Construction techniques may also be heuristic: identifying styles, patterns, or other idioms to apply in the synthesis of the view.

interpretation methods which guide readers to understanding and interpreting architecture views and their models.

analysis methods are used to check, reason about, transform, predict, and evaluate architectural results from this view, including operations which refer to model correspondence rules.

implementation methods are the means by which to design and build systems using this view.

Another approach to categorizing operations is from Finkelstein et al. [2]. The *work plan* for a viewpoint defines 4 kinds of actions (on the view representations): *assembly actions* which contains the actions available to the developer to build a specification; *check actions* which contains the actions available to the developer to check the consistency of the specification; *viewpoint actions* which create new viewpoints as development proceeds; *guide actions* which provide the developer with guidance on what to do and when.

7 Correspondence rules

⋆ Document any correspondence rules defined by this viewpoint or its model kinds.

Usually, these rules will be across models or across views since, constraints within a model kind will have been specified as part of the conventions of that model kind.

See: ISO/IEC/IEEE 42010, 4.2.6 and 5.7

8 Examples (optional)

Provide helpful examples of use of the viewpoint for the reader (architects and other stakeholders).

9 Notes (optional)

Provide any additional information that users of the viewpoint may need or find helpful.

10 Sources

⋆ Identify sources for this architecture viewpoint, if any, including author, history, bibliographic references, prior art, per ISO/IEC/IEEE 42010, 7e.

The template ends here!

References

Clements, Paul C. et al. *Documenting Software Architectures: views and beyond*. 2nd. Addison Wesley, 2010.

Finkelstein, A. et al. "Viewpoints: a framework for integrating multiple perspectives in system development". In: *International Journal of Software Engineering and Knowledge Engineering* 2.1 (Mar. 1992), pp. 31–57.

Hilliard, Rich. "ISO/IEC 42010 née IEEE Std 1471". In: *Documenting software architectures: views and beyond*. Ed. by Paul Clements et al. 2nd. Addison Wesley, 2011, pp. 400–405.

— "Viewpoint Modeling". In: *First ICSE Workshop on Describing Software Architecture with UML*. Position paper. May 2001.

IEEE Std 1471, IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. Oct. 2000.

ISO/IEC/IEEE 42010, Systems and software engineering — Architecture description. Dec. 2011, pp. 1–46.

Contents