IEEE Std 1471 and Beyond*

Rich Hilliard
ConsentCache, Inc.
rh@ConsentCache.com

January 1, 2001

Overview

I describe the key contributions of IEEE Std 1471 to the discipline of software architecture representation. After reviewing the contributions of IEEE 1471, I discuss how we (the community interested in Software Architecture) may build upon the foundation provided by IEEE 1471 to continue to improve and disseminate techniques for architectural description.

(Although three pages is insufficient to give a useful example of an IEEE 1471-conformant architectural description, there are a number of applications of IEEE 1471 in the literature. Visit the IEEE Architecture Working Group web site (http://www.pithecanthropus.com/~awg) for links.)

IEEE Std 1471 ...

IEEE Std 1471–2000 is IEEE's *Recommended Practice for Architectural Description of Software-Intensive Systems* [7]. To my knowledge, this is the first formal standard to address what is an architectural description (AD). It was developed by the IEEE Architecture Working Group with representation from industry, other standards bodies and academe, and was subject to intensive reviews by over 150 international reviewers, before its publication this past Fall.

IEEE 1471 establishes a set of *content requirements* on an architectural description (AD) – a collection of products to document an architecture. As such, the Standard plants a stake on how ADs should be organized, and their information content, while: (**i**) abstracting away from specific media (text, HTML, XML); (**ii**) being method-neutral (it is being used with a variety of existing and new architectural methods and techniques); and (**iii**) being notation-independent, recognizing that many diverse notations are needed for recording various aspects of architectures.

*Position paper for SEI's First Architecture Representation Workshop, 16–17 January 2001.

It achieves this by being based upon a conceptual framework for architectural description (see figure 1). The breadth of this framework is worth appreciating relative to current work in architectural research and practice. To my mind, much of this work has focused on what are portrayed as Models in the conceptual framework, including architecture description languages, and related tools. While important, much of this work lacks a larger context needed in most practical, industrial-strength applications. By reifying notions like Stakeholders and Concerns, the IEEE 1471 framework suggests a basis for dealing with these wider issues in a theory of architectural description.

Content Requirements on ADs

The content requirements of IEEE 1471 are stated in the terminology of the conceptual framework. These requirements define what it means for an architectural description (AD) to conform to the Standard. The principles underlying these requirements are briefly summarized here.

ADs are interest-relative: The audiences for an AD are the various stakeholders of the system, each with specific concerns (such as security, performance, or constructability) for the architecture. An AD should be explicit in addressing these stakeholders. Therefore, an AD must explicitly identify the system's stakeholders and their concerns for the system.

Concerns form the basis for completeness: An AD must addresses all stakeholders' concerns. If it does not, it is by definition, incomplete.

Multiple views: An AD is organized into one or more views. Each view is a representation of the entire system of interest intended to address a particular set of stakeholder concerns.

Although the use of views is hardly new with IEEE 1471, its contribution is to motivate the use of views (the source of much hand-waving in the Software Architecture literature) with respect to addressing specific concerns of specific stakeholders.

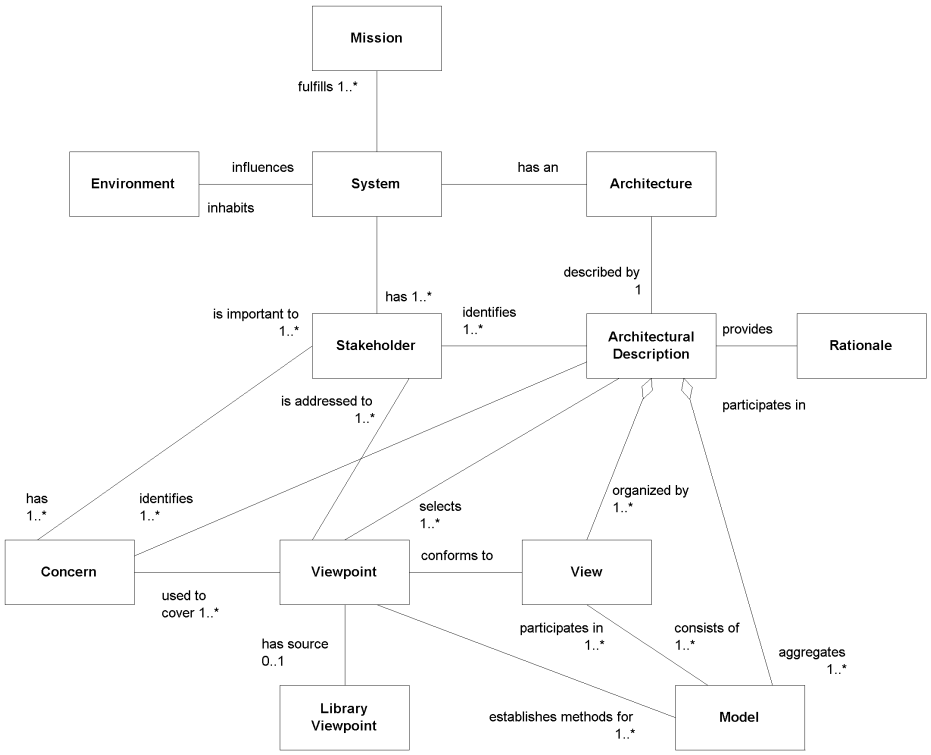

Figure 1: IEEE 1471 Conceptual Framework

Views are modular: A view may consist of one or more architectural models.

To satisfy the concerns to be addressed by a particular view, multiple notations may be used. This is one of the several places where IEEE 1471 is "parameterized" to accommodate the wide range of best practices in Software Architecture modeling.

Inter-view consistency: An AD must document any known inconsistencies among the views it contains.

This is a fairly weak requirement – based on current consensus; I imagine as a community we can do much better in the future (see below).

Views are well-formed: Each view has an underlying viewpoint identifying a set of architectural concerns and specifying how the architectural description meets those concerns, using languages and notations, models, analytical techniques and methods. A *viewpoint* is a set of conventions for constructing, interpreting and analyzing a view.

This is another "parameter" in IEEE 1471. Organizations may define and select their own set of useful viewpoints. In fact, IEEE 1471 does not even specify a fixed set of viewpoints; the Standard is "agnostic" about where viewpoints come from. Instead, the following principle is employed:

Concerns drive viewpoint selection: Each identified stakeholder concern must be addressed by one of the selected viewpoints.

Viewpoints are first-class: Each viewpoint used in an AD is "declared" before use (either "in line" or by reference). A viewpoint declaration establishes the stakeholders addressed by the viewpoint; the stakeholder concerns to be addressed by the viewpoint; the viewpoint language, modeling techniques, or analytical methods used therein; and the source, if any, of the viewpoint ("prior art"). A viewpoint may also include: any consistency or completeness checks associated with the underlying method to be applied to models within the view; any evaluation or analysis techniques to be applied to models within the view; and any heuristics, patterns, or other guidelines which aid in the synthesis of an associated view or its models.

This principle is perhaps the primary contribution of IEEE 1471 – to provide a means by which the many architectual techniques in use today may be uniformly described so that they may be used by others, compared and combined.

... And Beyond

In addition to codifying best current practices in architectural description, a goal of the IEEE for the develop-

ment of IEEE 1471 was to provide a foundation for the continuing evolution of the discipline of Software Architecture. To conclude this position paper, I briefly note a few opportunities of this kind.

Reuse. Viewpoints, being system-independent, are highly reusable. The viewpoint construct is intended to facilitate capture of one important kind of architectural knowledge: *when to apply given representational mechanisms to address particular stakeholder concerns* [5]. However, very little of present architectural knowledge is captured in this fashion. For example, there is much work in the academic literature on modeling architectures via components, ports, connectors, roles, and their configurations which might be termed a "Structural Viewpoint." By having a clear viewpoint declaration, it would be easier to apply this knowledge more uniformly. One useful role for organizations like SEI would be to serve as a repository for reusable viewpoints.

View Checking. IEEE 1471 is essentially silent on the issue of checking or analysis of individual views, except to say that a view must be well-formed with respect to its viewpoint – delegating the checking to any technique associated with the viewpoint.

Viewpoints will vary in their rigor, associated analytic techniques, etc., which may be brought to bear on checking a view. By having uniform declarations it may be possible to "lift" techniques developed for one notation to use with others. See [2] for a discussion of this in the context of use of the various notations of UML.

View Integration and Inter-view Consistency. It has been long recognized that introducing multiple views into architectural descriptions leads to an integration problem – how does one keep views consistent, non-overlapping?

> Complex specifications require structure, such as different segments for different concerns. However, different concerns also lead to different notations. ... [T]his leads to a *multiple-view problem*: different specifications describe different, but overlapping issues. [8] [my emphasis]

The introduction of viewpoint declarations, while not solving the problem, gives us a tool for detecting overlaps and inconsistencies, and potentially a substrate for solving the integration problem. See [3], [4], [1] for three different suggestions for tackling the view integration problem.

Formalization. The conceptual framework of IEEE 1471 is an informal, qualitative model. If it is useful, which appears to be the case, it may be insightful to attempt to formalize the concepts therein. Such a formalization could have benefits in several of the topics just mentioned: viewpoint reuse, view checking, view integration, and inter-view analysis.

Finally, there are another set of advanced topics in architectural description barely addressed by today languages and tools. See [6] for discussion.

References

[1] Alexander Egyed and Rich Hilliard. Architectural integration and evolution in a model world. In Bob Balzer and Henk Obbink, editors, *Proceedings Fourth International Software Architecture Workshop (ISAW-4), 4 and 5 June 2000, Limerick, Ireland*, pages 37–40, 2000.

[2] Rich Hilliard. Using the UML for architectural description. In Robert France and Bernhard Rumpe, editors, *≪UML≫'99 The Unified Modeling Language, Second International Conference*, volume 1723 of *Lecture Notes in Computer Science*, pages 32–48. Springer, 1999.

[3] Rich Hilliard. Views and viewpoints in software systems architecture. Position paper from the *First Working IFIP Conference on Software Architecture*, San Antonio, 1999.

[4] Rich Hilliard. Views as modules. In Bob Balzer and Henk Obbink, editors, *Proceedings Fourth International Software Architecture Workshop (ISAW-4), 4 and 5 June 2000, Limerick, Ireland*, pages 7–10, 2000.

[5] Rich Hilliard. Three models for the description of architectural knowledge: Viewpoints, styles, and patterns. Submission to WICSA-2, January 2001.

[6] Rich Hilliard and Timothy B. Rice. Expressiveness in architecture description languages. In Jeff N. Magee and Dewayne E. Perry, editors, *Proceedings of the 3rd International Software Architecture Workshop*, pages 65–68. ACM Press, 1998. 1 and 2 November 1998, Orlando FL.

[7] IEEE. *Recommended Practice for Architectural Description of Software-Intensive Systems*, October 2000.

[8] Mary Shaw and David Garlan. *Software Architecture: Perspectives on an emerging discipline*. Prentice Hall, 1996.