Architecting Processes are Key to Software Quality

Leon Osterweil
University of Massachusetts
Amherst, MA 01003 USA
413/545-2186
ljo@yquem.cs.umass.edu

ABSTRACT

In this position paper we explain why Architecting Processes, namely processes for using architectures to develop software, should receive more attention. Architectures are viewed as software engineering artifacts that can be used as the basis for developing superior software products. But the development of these products should still be accomplished with the guidance of processes. The processes are themselves software artifacts that should be developed in such a way as to demonstrably assure that they achieve their goals and requirements. Architecting Processes should be developed to demonstrably assure that software products are of high quality. Different processes will have different effectiveness in assuring this. Different architecture definition formalisms will be of differing value in supporting these different processes. The selection of an architecture definition formalism should, therefore, be strongly determined by the choice of Architecting Processes, and the goals and requirements that they must satisfy.

Keywords

Software process, software architecture, software quality

1 INTRODUCTION

There should be greater interest in the goals and purposes of architecture. Currently most interest seems to be focussed on the artifacts that comprise a software architecture. There is considerable difference of opinion on what the nature and variety of these artifacts should be. We believe that disagreements on this are hard or impossible to resolve in the absence of a clear sense of what the purpose of an architecture is. We believe that different architecture artifacts and representation formalisms are of differing value in support of different goals and purposes. It seems agreed that good software architectures can facilitate reuse, support evolution, help in detecting serious errors early, and generally help assure the quality of software products. But it is far from clear that any given architecture representation will be most effective in supporting all of these goals. Thus, we suggest that more attention must be addressed to the question of what architectures are for, then which architecture representation formalisms are effective in supporting which of the various goals for architectures, and then how those representation formalisms can be used most effectively to support the achievement of those goals.

The activities entailed in pursuing understood, specified goals for an architecture are what we refer to as "Architecting Processes." It seems clear that superior architecting processes can be of enormous importance when integrated effectively into overall software development processes, and are themselves software artifacts. In earlier work we have suggested that "Software processes are software too" [Osterweil 87, Osterweil 97]. That being the case it is useful to consider the requirements, design, and implementation aspects of Architecting Processes. From this perspective, we see that the goals and purposes of creating architecture artifacts are in fact Architecting Process requirements. Specifications of the orderly, systematic way in which one goes about achieving those goals are what we refer to as Architecting Processes models and code. Thus, our position is that the current preoccupation with formalisms for specifying architecture artifacts should be better balanced by increased attention to the issue of developing Architecting Processes that use them. We suggest that the study of Architecting Processes can provide clear evidence of which architecture representations can be used in which ways to effectively support the achievement of which architecting goals. Further, the study of Architecting Processes should also lead to understandings of the ways in which existing architecture representations could be improved to make them more useful.

As just noted, process development must begin with specification of process requirements, and thus our investigation of Architecting Processes should begin with identification of architecting requirements. It is our belief that one of the key goals and purposes of architecting is to assure the quality of product software that is developed from the architecture specified. Thus, to demonstrate the value and applicability of our ideas we begin by establishing as a goal the development of application software that is of demonstrably high quality. We then indicate how Architecting Processes can help in achieving that goal, and how consideration of these processes can provide indications of which architecture specification formalisms provide the most effective support for processes that contribute most strongly towards that goal.

There are a number of types of processes that can contribute towards using architectures to achieve software product quality. For example, some types of Architecting Processes can be used to assure that the architecture developed is sound, well-formed, and in keeping with product software objectives. To these processes, the architecture itself is an object of study and analysis and the processes examine it to assure that it is indeed a reliable guide to effective development. Other types of processes, presumably executed subsequently, are used to assure that the application software that is developed is indeed developed in accordance with the architecture specification. This then assures that the validated architecture is used faithfully as a true guide in the implementation of software that is then reliably known to have the quality characteristics and properties that are desired. Still other types of processes analyze explicit, formal representations of Architecting Processes such as those that we have just described, in order to determine how well these Architecting Processes seem to support the achievement of state software product quality objectives.

2 DETERMINING ARCHITECTURE QUALITY

As observed above, we believe that an architecture should be considered to be an artifact, and that it is important to be able to assure that it is of high quality. There are a variety of different characteristics and properties that a high quality architecture should be expected to possess. Among them are consistency, completeness, responsiveness to requirements, and implementability. Clearly some (perhaps all) of these are difficult or impossible to determine definitively. On the other hand it is reasonable and important to consider that they can and should be determined increasingly definitively with the help of effective processes.

Thus, for example, the consistency of an architecture specification can and should be studied. For example all specified dataflows should be specified consistently, and every hierarchical elaboration should be consistent with the specification that it is elaborating. It is not difficult to imagine the outlines of processes that are able to make these kinds of consistency determinations, although the specific details may vary considerably with the formalism in which the architecture specification is expressed.

Processes for determining the degree to which an architecture is responsive to requirements are harder to conceive of. In particular, in this case it is not clear how one would quantify the degree of responsiveness. Processes that have this goal are more likely to need to be considered to be iterative and open-ended.

Thus the spectrum of types of processes for determining Architecture Quality is quite broad. Furthermore, their effectiveness clearly depends in important ways upon the kinds of semantic details that are expressible in the architecture specification formalism itself. These observations seem to reinforce our contention that it is important to clearly and precisely state the goals and requirements for processes. In this case, these goals and requirements must address the dimensions and quantifications of architecture quality that are to be achieved. These in turn should presumably be derived from corresponding software product quality goals and requirements. Once these have been established, processes for determining them can be devised. But the degree to which such processes can be effective may often be determined by the choice of architecture formalism.

This underscores our contention that the choice of architecture formalism should be powerfully molded by consideration of the Architecting Processes needed in order to meet product quality goals and requirements.

3 ASSURING THAT IMPLEMENTATIONS CONFORM TO ARCHITECTURES

Other processes seem important in determining how well actual implementations adhere to the guidance and specifications of the architecture. Here too, the question of how to determine and quantify the notion of conformance is not an easy one, as it has many facets. Thus, it is important to determine, for example; that an implementation achieves the functionality that an architecture specifies, that the implementation's structure is as specified by the architecture, and that the modularization, communication patterns, and control structures in the implementation are those that the architecture has specified. Indeed there are virtually limitless ways in which it might be important to compare the actualities of an implementation to the specifications of an architecture. For each way, it is possible to imagine a variety of ways in which the comparison can be done.

Here too, we see that there is a diversity of processes that might be employed to make a particular type of determination of conformance of implementation to architecture. There is a diversity of types of conformance that might be desired as well. The overall process for determining the degree to which an implementation conforms to its architecture is some sort of synthesis of these component processes. These processes are also Architecting Processes. They are Architecting Processes that are aimed at demonstrating that the architecture (presumably one that has already been shown to be of high quality by earlier Architecting Processes) has indeed been faithfully used as a guide to the actual implementation. We might refer to them as Architecture-Based Implementation Verification processes.

Here too, we note that any particular Architecture-Based Implementation Verification process used to compare an implementation to its architecture should be created in response to clearly understood and stated implementation

evaluation goals and requirements. These goals and requirements should be derived from overall product quality goals and requirements. These Architecting Processes must be designed to demonstrably deliver the product quality assurances that are required. Different architecture specification formalisms will support different of these processes to differing degrees. Thus, here too we see that the choice of architecture formalism should be strongly guided by these considerations.

4 COMPARISON AND EVALUATION OF ARCHITECTING PROCESSES

In the previous two sections we have argued for clearer recognition of the importance of Architecting Processes. While the attention of the software architecture community is currently focussed primarily on software architecture artifacts, there is a clear implicit recognition that these artifacts are developed to be used for some software product related purposes. We have emphasized that software product quality is one such important purpose, and have indicated how Architecting Processes are vehicles for using architecture artifacts to achieve software product quality.

In these previous sections we have also emphasized that these Architecting Processes are software that can and should be developed to meet goals and requirements that need to be made as clear and explicit as possible. It now seems important to note that this implies that it is also important to determine the quality of Architecting Processes themselves. This quality is essentially the degree to which the Architecting Processes conform to the requirements that should have driven their development. Thus, for example, if clean modularization of a software product is a requirement, then Architecting Processes should be used to demonstrate that the architecture driving the implementation defines a clean modularization and that implementation of the architecture adheres to the specified modularization. Architecting Processes that make no attempt to do either or both are then clearly poor quality processes. Architecting Processes that incorporate effective techniques for determining this characteristic are of relatively higher quality.

We strongly advocate the use of process definition and specification formalisms as the basis for definitively making these process quality determinations. When these process specifications are precisely defined using rigorous formalisms, it is possible to adduce definitive arguments about their quality--namely the degree to which they support the achievement of product quality objectives.

It is important to note that process specifications must incorporate specifications of the products upon which they operate. In the case of Architecting Processes, the primary artifact with which they are concerned is the architecture specification. Thus we have just also argued for the importance of rigorously defined architecture specifications. As noted above, the particular architecture specification formalism should be chosen to match the Architecting Process that will use it.

5 SUMMARY

In summary, we note that 1) different product quality objectives are supported by different processes, 2) the quality of these processes can be demonstrated best through argumentation about Architecting Process, 3) different architecture formalisms are more or less suitable for use by different Architecting Processes, and 4) different formalisms are better vehicles than others for supporting various of the objectives and demonstrations.

In short, architecture specification formalisms can and should be evaluated in terms of how well they support the specification of Architecting Processes that are demonstrably superior in helping to meet specified software product quality goals.

REFERENCES

[Osterweil 87] L.J. Osterweil, *Software Processes are Software Too*, Proc. 9th Intl. Conference on Software Engineering, Monterey, CA, IEEE Press, 2-13.

[Osterweil 97] L.J. Osterweil, *Software Process are Software Too, Revisited*, Proc. 19th Intl. Conference on Software Engineering, Boston, MA, ACM Press, 540-548.