

Layered Architecture Revisited – Comparison of Research and Practice

Juha Savolainen and Varvana Myllärniemi
Nokia Research Center, Helsinki University of Technology
Juha.Savolainen@nokia.com, Varvana.Myllarniemi@tkk.fi

Abstract

Organizing a software architecture into layers has been one of the earliest architectural styles ever used. Even today layered structure is a very common architectural style used in various industrial systems. However, we have observed that the usage of layered architectural style varies greatly in different contexts. This paper aims to compare the notion of software architecture layers in research literature as well as in industrial practice. Firstly, we performed a systematic literature review of research articles on layered software architectures; we also reviewed selected books of software architecture. Secondly, to understand the practice, we investigated a number of different recent architecture documents to cover the current usage of layered architectures. Our results indicate that there is very little actual research done on layered architectures. The current usage of layered structures seems to be more complex than reported before. This gap between the research and practice needs to be bridged by researchers.

1 Introduction

Organizing a software intensive system into layers has been one of the earliest architectural styles ever used to design the system architecture. Most papers refer to Dijkstra [1] as the first usage of hierarchical organization of software system. Also [2] elaborates this hierarchy further. Since then the use of layers have been reported in numerous books (e.g. [3,4]) and seems to be very established. Still today layered structure seems to be a very common architectural style used in various industrial systems.

However, we have observed that the usage of layered architectural style varies greatly in different contexts. This raises a question on what is the common research foundation on which the industrial practice rests? Has the usage of layers evolved since its introduction in 1960s and 1970s?

To understand current research knowledge, we performed a systematic literature review to find out what is the current research understanding on layered architectures. We also reviewed the definition and usage of layered architectures in selected publications and books of software architecture and design. Hence,

the research methodology used for this research objective was systematic literature review as described by Kitchenham [5]; this was augmented with additional references.

To understand the practice, we investigated 5 different recent architecture documents to cover the current usage of layered architectures. In these documents we tried to find examples of the usage of layer diagrams that exceeds or contradicts the established understanding on layered architectures. Hence, the research methodology used for this research objective was a very light-weight qualitative case study [6] that used documents as its main data source.

Our results indicate that there is very little actual research done on layered architectures. There have been very few conference or journal papers on the foundation of layered architectures, most of the books seem to either be based on industrial practice or on the same seminal works [1,2]. Our findings on the architectural documents imply that the current usage of layer diagrams seems to be more complex than reported before.

Especially, practitioners use many layer diagrams to describe various aspects of one system. These several layer diagrams seem to place constraints on the existing architecture and development of the subsystem architectures. That is, layer diagrams can have elements in various maturity levels.

Having multiple layer diagrams places many constraints on the architectural elements. Having these constraints only in one abstraction level seems to be complicated. Therefore, it seems that in practice the architectural elements are described in many different granularity and abstraction levels. All these elements can then participate in single layered diagram.

The remainder of this paper is organized as follows. Section 2 introduces the literature review and discusses its findings. Section 3 describes the found usage of layered architecture in industry. Finally we discuss our findings and conclude.

2 Literature review

2.1 Systematic literature review

The aim of the systematic literature review was to study the research literature on software architecture

layers. In the following, a brief description of the study steps and results is given.

The phases of the study followed the guidelines of Kitchenham [5]. Firstly, the study was planned, that is, the need for the literature review was established as well as initial study protocol developed. Secondly, literature review was conducted. Finally, the results were written in this paper. In the following, we will present the steps of conducting the review [5] in more detail.

Identification of research. The review protocol that was used for identifying primary studies was based on finding primary studies covering layers in software architecture. The primary studies were searched from five established databases containing scientific publications. Table 1 identifies the used search strings as well corresponding databases. The titles and abstracts of found papers were downloaded and imported to EndNote program.

Table 1 Identification of research

IEEEExplore	81 papers
(((software<in>ti)<or>(software<in>ab)) <and> ((architecture<in>ti)<or>(architecture<in>ab)) <and> ((layer<or>layered)<in>ti))	
ACM Digital Library	65 papers
(Title:software* or Abstract:software*) and (Title:architecture* or Abstract:architecture*) and (Title:layer*)	
Science Direct	21 papers
title-abs-key((software) AND (architecture*)) AND title(layer*)	
ISI Web of Science	64 papers
Topic=(software*) AND Topic=(architecture*) AND Title=(layer*)	
SpringerLink	15 papers
(ab:(software*) or ti:(software*)) and (ab:(architecture*) or ti:(architecture*)) and ti:(layer*)	

Selection of primary studies. In total, the search over databases as described in Table 1 yielded 246 papers. In order to find relevant literature from this set, we utilized the following inclusion and exclusion criteria in selecting primary studies.

The paper was included if it:

- Was in the field of software engineering / computer science AND
- Covered layers in software architecture AND
- Defined, evaluated, verified, analyzed, or modeled layers OR
- Discussed requirements, constraints, or limitations to layers

The paper was excluded if it:

- Covered hardware or computer architecture

layers OR

- Covered telecommunication protocol layers OR
- Covered only one layer (e.g. in a larger layered framework)

The process of including and excluding papers is as follows. Firstly, duplicates were removed with the aid of EndNote as well as manually. Thereafter, inclusion and exclusion criteria were applied to papers based on their title and abstract. Thereafter, selected papers were downloaded and read, and the inclusion and exclusion criteria were applied based on paper content (mainly introduction and conclusions). As a result, this step yielded 11 primary papers [8-18].

Study quality assessment. Kitchenham [5] advocates the use of study quality assessment to remove the effect of low-quality studies on results. However, given the small number of primary studies as well as the overall methodological weakness of software engineering papers, we did not wish to exclude papers based on their quality. This phenomenon was also acknowledged by Staples & Niazi [7], who found the strict guidelines to be too restrictive in the study quality assessment.

Data extraction and synthesis. Data was extracted from the papers with the aid of the following questions.

- 1) What is the contribution of the paper?
- 2) What is considered as a layer?
- 3) What relations does layering set between layers?

Results. Although there exist many scientific papers that cover layers in some form or another, very few papers actually discuss the notion and concepts of layers. Most papers found from databases with search strings applied layers in some form or another. Typical applications were in the domain of internet and telecom protocols.

When synthesizing the notion of layers in selected primary papers, it can be said that they are relatively diverse. Some are very specific (for example, discuss layers as sets of source code files), whereas some are more generic and define layers as sets of elements in the architecture. Some discuss layers as in distributed communication systems, whereas some discuss layers in object-oriented systems. The relations between layers stem from this notion of layers.

However, defining layers as sets of elements does not reveal the full picture: also the principles for decomposing and grouping elements into layers should be discussed. Some papers are relatively vague in this respect. Most agree that layering should follow some kind of separation of concerns. Many papers mention levels of abstraction as one guiding principle. There are also a couple of papers that discuss layering as a means of separating reusable parts in software architectures,

that is, re-use based layering. Herzberg [15] distinguished between layering as a principle of using abstraction layers to understand a system by levels of abstraction and layering as a principle of structuring a system architecture in layers of services and functionality.

To summarize, the systematic literature review revealed the lack of scientific publications that describe and define the concepts related to architectural layers.

3 The usage of layered architecture in Industry

In this section we discuss the usage of layered architectures found in the industry. Via case descriptions, we wish to highlight the intention, layer notion, relations between layers, and criteria for constructing layers.

In order to understand the current usage of layered architectures in the industry, we reviewed 5 different product or product line architecture documents. All these documents were in the telecommunications domain, comprising architecture documents in Nokia (mobile phones and Internet services) and Nokia Siemens Networks (telecommunications infrastructure).

The method for selecting the cases was mainly convenience sampling [6]: these were the projects where we had easy access to the documents. Hence, we do not claim that the set of architecture documents is representative or complete. However, the cases do give insight into the industrial usage of layers.

Three architecture documents described very traditional way to apply layers. They used them as the way to describe hierarchical organization of software. Two of these used this description mainly in informal way without strict semantics. One of the systems clearly had a rationale to limit dependencies between the different abstraction levels of the software.

Two more interesting systems had more complex usage of layered architectural view. First system used metamodels to define allowed architectural elements and their valid usage. Layers were an explicit part of the metamodel. While this clearly represents more advanced usage of layered view, it was reasonably similar to the usage reported by other sources, e.g. [19]. Therefore, it is not further elaborated in this paper to save space.

Here we will focus only on one out of the five systems studied. It is a base station platform project that intended to create a new platform architecture that would allow reusing the basic functionality over various base station products supporting different radio access protocols.

The case discussed had two different layered diagrams that place constraints on the software structure. Since they operate on the same structure, constraints implied by both diagrams must be fulfilled by the software.

None of the reported cases in research considered the possibility of having more than one layered structure on the same set of components. Having only one layered structure is quite natural if the criterion for structuring the software in layers is mainly driven by obtaining different abstraction levels. The idea of virtual machines tends to promote having only one layered structure.

However, when we add reuse-based layering, one can possibly have many dimensions. In theory, a software architecture could have a nearly unlimited number of layers intending to separate software into specific and generic parts for each dimension of variation. However, more typical is to align reuse-based layering to variations in respect of products. Then the bottom layer is formed by all modules that are shared by all products of the product line.

In the base station architecture, combining both reuse based layering and virtual machine based layering produces a complex structure and limitations on the dependencies. This is made even more complex by the fact that the component decomposition criteria were driven by functionality and the area of expertise as discussed earlier.

Having different criteria between the overall decomposition and layering leads to situation where software elements cannot be mapped to the layers in single level of granularity. This has implications also in the development process. Layering must be considered not only for the overall architecture, but also when designing the next levels of decomposition.

Having multiple layered structures highlights the interpretation of layered architecture through the constraints that is places on software. It views the layering as a tool to organize and visualize constraints on the software dependencies.

4 Discussion and conclusions

In this paper, we argued that any number of layering diagrams for one software architecture is theoretically possible, given that the constraints they are placing are not contradicting. Also, even the decomposition criteria can be different in each layer diagram as well as for the whole system structure.

Based on our experience and 5 reviewed architectures, most architectures do not take the advantage of the flexibility achieved by using multiple layering criteria at the same time. We believe that this is so mainly for three reasons.

Firstly, none of the books or scientific publications studied in this paper addresses the question of multiple layered diagrams. Hence it is unlikely that practitioners take advantage of such structure. Secondly, having multiple criteria for layering makes understanding of the architecture more difficult. One cannot have just one layered diagram, rather than a set of diagrams that together form the layered view. Finally, one must typically have software elements on different levels of abstraction in the layered diagrams. This further complicates understanding of the software. In addition, this may require taking account the constraints placed by layers in different stages of the software development lifecycle.

Our findings indicate that the current usage of layered architecture is more complex than what have been reported in the literature. Even with a limited study we managed to find one case that exceeds the current state of research. Further, the lack of research on the topic of this key architectural style is surprising. When combining the results of the systematic literature review and our practical architecture document – one can question the foundation on top of which layered architecture research and practice has been built.

What is the real definition of the layered structure? Should research follow practice and define more complex and flexible definition for layered architectures? Or should the structure presented in real architectures be called something else?

And finally, if the findings of this study are true for layered style that is arguably one of the most well know architectural tools, what does this mean for other styles? Are e.g. pipes and filters used currently as described in software architecture publications?

In future, we intend to enlarge the sample of architecture document in the industry to get even better view how layers are used. We also intend to perform more complete survey on books that discuss layered architectures, since they seem to be a better source for capturing the current research understanding on layered architectures.

5 References

- [1] Dijkstra, E. W. The structure of the “THE”-multiprogramming system. *Commun. ACM* 11(5). May. 1968.
- [2] Parnas, D. L. On the criteria to be used in decomposing systems into modules. *Commun. ACM* 15(12). Dec. 1972.
- [3] Shaw M, Garlan D. *Software Architecture — Perspectives on an Emerging Discipline*, Chapter 2. Prentice Hall, 1996.
- [4] Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Robert Nord, and Judith Stafford. *Documenting Software Architectures: Views and Beyond*. Addison-Wesley, 2002.
- [5] Kitchenham, B.. Procedures for performing systematic reviews. Technical report TR/SE0401, Keele University. 2004.
- [6] Patton, M. Q. *Qualitative evaluation and research methods* (2nd ed.). Sage Publications, 1990.
- [7] Staples, M. and Niazi, M. Experiences using systematic review guidelines. *Journal of Systems and Software*, 80(9). 2007.
- [8] Sarkar, S., G. M. Rama, et al. A Method for Detecting and Measuring Architectural Layering Violations in Source Code. *Software Engineering Conference*, 2006. APSEC 2006. 13th Asia Pacific. 2006.
- [9] Park, C., E. Hong, et al. A method to preserve layered architectural style in development phases. *Ieice Transactions on Information and Systems* E87D(7): 1965-1970. 2004.
- [10] Lague, B., C. Leduc, et al.. An analysis framework for understanding layered software architectures. *Program Comprehension*, 1998. IWPC '98. Proceedings., 6th International Workshop on. 1998.
- [11] De Paoli, F. and A. Sosio. Requirements for a layered software architecture supporting cooperative multi-user interaction. *Software Engineering*, 1996., Proceedings of the 18th International Conference on. 1996.
- [12] Myllymaki, T., K. Koskimies, et al. Structuring product-lines: A layered architectural style. *Object-Oriented Information Systems*, Proceedings of the 8th International Conference on. 2002.
- [13] Paris, M. Reuse-based layering: a strategy for architectural frameworks for learning technologies. *Advanced Learning Technologies*, 2004. Proceedings. IEEE International Conference on. 2004.
- [14] Herzberg, D. and A. Marburger. The use of layers and planes for architectural design of communication systems. *Object-Oriented Real-Time Distributed Computing*, 2001. ISORC - 2001. Proceedings. 2001.
- [15] Herzberg, D. and M. Broy. Modeling layered distributed communication systems. *Form. Asp. Comput.* 17(1): 1-18. 2005.
- [16] Snoeck, M., S. Poelmans, et al. A layered software specification architecture. *Conceptual Modeling - ER 2000*, 19th International Conference on. 2000.
- [17] Spicer, K. L. A successful example of a layered-architecture based embedded development with Ada 83 for standard-missile control. *Ada Lett.* XX(4): 50-63. 2000.
- [18] Kim, J. and C. R. Carlson. Design units - a layered approach for design driven software development. *Information and Software Technology* 43(9): 539-549. 2001.
- [19] Hofmeister, C., Nord, R. and Soni, D. *Applied software architecture*. Addison-Wesley, 2000.