The Journal of Systems and Software 99 (2015) 97–108

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

# Modelling large-scale information systems using ADLs – An industrial experience report

Eoin Woods [a,*], Rabih Bashroush [b]

[a] Artechra, Hemel Hempstead, UK
[b] School of Architecture, Computing and Engineering, University of East London, London E16 2QN, UK

ARTICLE INFO

*Article history:*
Received 22 May 2013
Received in revised form 4 September 2014
Accepted 15 September 2014
Available online 28 September 2014

*Keywords:*
Architecture description language
Software architecture discovery
Industrial experience report

ABSTRACT

An organisation that had developed a large information system wanted to embark on a programme that would involve large-scale evolution of it. As a precursor to this, it was decided to create a comprehensive architectural description to capture and understand the system's design. This undertaking faced a number of challenges, including a low general awareness of software modelling and software architecture practices. The approach taken by the software architects tasked with this project included the definition of a simple, very specific, architecture description language. This paper reports our experience of the project and a simple ADL that we created as part of it.



## 1. Introduction

There has been a great deal of academic and some industrial research into the definition of architecture description languages (ADLs) to assist with the difficult task of clearly defining the architecture of software intensive systems and there is still a significant amount of such research underway today (Di Ruscio et al., 2010; Cuenot et al., 2010; Oquendo, 2004). However, there is limited evidence of significant industrial use of the ADLs that have been produced, which we believe is for a number of reasons (Bashroush, 2006; Woods and Hilliard, 2005) including the narrow focus of most ADLs and the mismatch between their strengths and the needs of practitioners. This is particularly marked in the information systems domain, where it is difficult to find any large-scale use of ADLs, whereas there has been some documented use of ADLs in embedded and real-time systems (Oquendo, 2004; van Ommering et al., 2000; Allen et al., 2002).

In this paper, we report on the experience gained from the creation of a large architectural description for a complicated information system, in an environment where there was no existing use of UML, SysML or specialist ADLs and where it was felt that such approaches would not be successful. We describe the experience of that project, which was used as an opportunity to explore the use of a simple, domain specific, architecture description notation in an industrial context.

This paper explains the context of the project and the work undertaken during it, including the definition of a simple graphical notation and the experience of using the ADL with software development teams to produce architecture description documents. We also reflect on the experience in order to identify the lessons learned and discuss why we did not attempt to reuse an existing ADL from the many that can be found in the research literature.

The specific contribution of this work is to describe the experience of creating a large industrial architectural description intended for long-term use and the factors that we found to be important in successfully achieving this. While we did create a specific notation and structuring approach for the project, this was a side effect of the project, not its goal, and our intention is not to contribute yet another general purpose ADL to the research literature. In fact, as we explain at the end of the paper, based on this specific experience, we concluded that general purpose ADLs might be less useful for industrial use than has been previously assumed; the ADL we created is described here merely to explain what we found to be effective in this project.

In the next section we present an overview of related work on ADLs in both industry and academia. Section 3 provides background information about the work and the context of the project. Section 4 then explained the rationale and drivers of the project. The approach used is described in Section 5. The ADL design, along with the system architectural style is presented in Section 6. A case study is then presented in Section 7. The experience and lessons learned from the project are discussed in Sections 8 and 9 respectively. Finally, Section 9 completes the paper with the summary and conclusion.

* Corresponding author. Tel.: +44 207 568 2764.
*E-mail address:* eoin.woods@artechra.com (E. Woods).

http://dx.doi.org/10.1016/j.jss.2014.09.018
0164-1212/

## 2. Related work

As explained in the previous section, this paper reports an industrial experience of applying ADL concepts to the description of a significant industrially developed information system. Directly related work would be other similar case studies and experience reports. On searching the research literature, we did not find any directly equivalent work, where an architectural description language was used to describe a large information system, although there have been published reports of ADLs being used to describe embedded or real-time systems (such as Feiler et al., 2000; Lonn et al., 2004).

However given that as part of this project we decided to create our own notation for the architectural description, it is worth considering related work in the ADL field and why we did not choose to reuse an existing ADL.

Over the past two decades, an increasing number of ADLs have been developed, largely within academia (Medvidovic and Taylor, 2000; Clements, 1996). Although some ADLs have been put to industrial use in specific domains (Cuenot et al., 2010; Oquendo, 2004; van Ommering et al., 2000; Standard, 2006), the majority of ADL projects remain confined to laboratory-based case studies.

While ADLs originated in academia such as ACME (Garlan et al., 2000)/ADML, ABACUS (Dunsire et al., 2005), Aesop (Allen, 1997), UniCon (Shaw et al., 1996), Wright (Allen and Garlan, 1996), GENVOCA (Batory and Geraci, 1997), π-ADL (Oquendo, 2004), Rapide (Luckham et al., 1995), SADL (Moriconi and Riemenschneider, 1997), xADL (Khare et al., 2001), ADLARS (Bashroush et al., 2005), ALI (Bashroush et al., 2008; Bashroush and Spence, 2009), ArchiMate (Lankhorst et al., 2009) and ByADL (Di Ruscio et al., 2010), to name a few, all exhibit novel approaches to architecture description, from support for interchange and interoperability to advanced architectural analysis capabilities, the vast majority tend to be vertically optimised, limiting their attractiveness in many industrial projects.

It is important to state that many of these ADLs probably *could* be used in an industrial context, but there is often no strong reason to do so. In general, academic ADLs focus more on analytical evaluation and rigour while in this project, and many other industrial projects, the focus was more on accessibility, practicality, and the ability to rapidly obtain a reasonably complete view of the structure and behaviour of the system. Those ADLs that do support the kind of description we wanted to create (such as ACME and xADL) are general-purpose languages that are not used in mainstream practice. Accordingly, they would have needed a lot of investment in tailoring and extension to fit our requirements, not to mention the tool support development effort (such as providing drawing support in standard tools). This meant the benefits we would have gained from using them did not appear to be large enough to justify the adoption overhead.

Considering these various factors together, our conclusion was that there was not a strong reason to adopt a research ADL for this work and we judged that it was going to be simpler and quicker to develop our own special-purpose notation.

The use of ArchiMate (Lankhorst et al., 2009) was also considered given the fairly wide spectrum it provides for enterprise architectural description. However, upon closer investigation, we found that the primitives in the ArchiMate language were not a particularly good fit given our need to describe system (i.e., software) architecture rather than enterprise architecture in this project.

As mentioned above, outside the area of information systems, there have been a number of industrial applications of ADLs for embedded and real-time systems, from consumer electronics (e.g., Koala van Ommering et al., 2000), π-ADL (Oquendo, 2004) to aeronautics and automotive systems (e.g., AADL Standard, 2006) and EAST-ADL (Cuenot et al., 2010). The use of ADLs in these application domains has enabled automated system analysis, and automated code generation (e.g., MetaEdit+ Smolander et al., 1991). However, given that such capabilities were less important for this project than our much simpler goals of easy adoption and straightforward system description, and the fact that these ADLs are specialised for embedded systems rather than large information systems, we did not feel that we should receive a return on the investment required to tailor and adopt them. We discuss our reasoning for not reusing an existing ADL further in Section 5.

## 3. Background to the work

This project was undertaken in a financial services firm that has developed a large custom information system to run its business. The software has been developed over a period of about 15 years and has grown from quite modest beginnings to the large system it is today, comprising millions of lines of code, storing several terabytes of information. The system includes software modules that have been developed from scratch within the organisation along with modules that have been acquired as a result of organisational acquisitions and that have been modified to integrate with the rest of the system.

Today, the system comprises about 20 major subsystems and over 10 million lines of Java, C++, C# and Perl, sharing a large multi-terabyte relational database. Although some members of staff who worked on the system in its early days are still with the firm (and actively involved with the system) it has grown to a size that means no individual understands it all, even at a reasonably high level of abstraction.

At the start of the project, there was no overall unified system description, although some teams responsible for subsystems did have their own documentation. This meant that the operation and interconnectedness of the system was often difficult to judge and this was starting to hinder change and evolution.

The organisation wanted to perform some wide ranging evolution and modernisation of the system's implementation and realised that a useful first step, to enable better intellectual control over the system, would be to capture a unified description of the system's architecture. This led to the project described in this paper being undertaken.

## 4. Overview of the project

The lack of a unified system description and a new initiative to modernise and restructure parts of the system led senior managers to initiate a project to "document" the system. At the outset it was not entirely clear what "document" meant, but discussion and exploration led to the conclusion that a current state architecture description was required (i.e., a description of the system's architecturally significant elements, responsibilities and interactions, rather than more detailed documentation of the design of individual modules).

A team of two experienced architects was tasked with this project, with a remit to define an approach and then work with the software development teams to create the architecture description.

One immediate complication was the lack of a clearly defined use for the documentation once it was available. A number of senior managers considered the creation of the documentation to be important, but it was not clear what they intended to use it for. Specifically, it was not clear if this was to be a living document, that the organisation aspired to keep current, or a snapshot to be used for planning, which would then be deliberately abandoned. The target audience also was not well defined, so we did not know whether it was to be a senior management planning tool or a more detailed description to be used by designers for tasks like impact analysis.