An Approach to Analysis and Design for COTS-Based Systems

Grace A. Lewis

Carnegie Mellon® Software Engineering Institute, Pittsburgh, PA, USA
glewis@sei.cmu.edu

Abstract. From an analysis and design perspective, developers of COTS-based systems face many challenges driven by built-in product paradigms as well as the volatility of the marketplace. One way to deal with these challenges is to adopt a spiral development process that allows for concurrent discovery and negotiation of user needs and business processes, applicable technology and components, the target architecture, and organizational constraints. This paper outlines a workflow for Analysis and Design that can be used within spiral-based development processes for building systems from commercial components.

1 Introduction

From an analysis and design perspective, developers of COTS-based systems face many challenges, especially in COTS-aggregate systems[1]:

- Product selections have dependencies on other products that need to be considered in the design.
- Product selections have built-in models of use and architectural paradigms that make integration difficult if this information is not available to the developers.
- The volatility of the marketplace and the frequency and content of product releases cause disruptions in the design process.
- Design decisions may have to be made with incomplete information because the products are treated as black boxes.
- Integration becomes a primary source of risk.

One way to deal with these challenges is to adopt a spiral development process that allows for concurrent discovery and negotiation of user needs and business processes, applicable technology and components, the target architecture, and organizational constraints. The Software Engineering Institute has defined a process framework called Assembly Process for COTS-Based Systems (APCS) and an instantiation of APCS called the Evolutionary Process for Implementing COTS-based systems (EPIC) [1] [3]. These are both spiral approaches to COTS-based systems (CBS) development.

[1] A COTS-aggregate system is a system composed of multiple COTS products from multiple vendors, integrated to collectively provide system functionality [4].

X. Franch and D. Port (Eds.): ICCBSS 2005, LNCS 3412, pp. 236 – 247, 2005.
© Springer-Verlag Berlin Heidelberg 2005

This paper outlines a workflow for Analysis and Design that can be used within CBS development processes such as EPIC. The workflow uses several techniques for building systems from commercial components to deal with the challenges outlined above [7].

2 The Analysis and Design Workflow

EPIC is a risk-based, disciplined, spiral-engineering approach to COTS-based systems development which leverages the Rational Unified Process (RUP). The element of interest for this paper is the Analysis and Design workflow. Workflows are sequences of activities that produce a result of observable value [5]. The goal of the Analysis and Design workflow in RUP is to produce an architecture and design that is refined and analyzed over the system's lifecycle. The main elements of the Analysis and Design workflow in RUP are system architecture, component design, and database design. These elements, although important in custom development, do not accurately represent the analysis and design process for COTS-based systems. Figure 1 presents a variation of this workflow that takes into consideration the nature of COTS-based systems[2].

In spiral development, all iterations have objectives and address certain risks. An objective of an early iteration, once the problem is better understood, should be the initial version of the system architecture and design. In subsequent iterations an objective might be a better understanding of technologies, evaluation of alternatives, or the selection of an alternative as the system architecture[3].

The development of the architecture of any software system usually starts with a box-and-line diagram where boxes represent system components and lines represent the interaction between components. The difference with CBS is that depending on what stage the project is in, these components can be technologies, products or versions. As the project advances and decisions are made, certain components will go from technology, to product, to version. On the other hand, if the use of certain products is an organizational mandate, components can start as versions. An example of a box-and-line diagram for a typical web application is shown in Figure 2.

When you put together a box-and-line diagram, you have an idea of what types of components you want to use and how they should be connected, but there are many unknowns about the components and the way these components interact. For example:

- What servlet engines and application servers are good choices for this web application?
- What application servers interact well with Oracle DB (organizational mandate)?
- What browsers have to be supported?
- Will all servlet engines work well with all browsers?
- What combinations work better?

---

[2] For readers not familiar with RUP, the Analysis and Design workflow is executed in every iteration that has a goal related to system architecture and design.

[3] The terms architecture and design will be used indistinctively throughout the paper. In CBS the line between the two is very blurry because components are treated as black boxes for which the detailed design is unknown; therefore a single artifact usually represents the architecture and design of a system.