Overlooked Aspects of COTS-Based Development

October 29, 2012
Goal

- Provide clear definition of commercial-off-the-shelf products.
- Empirically explore the subject to derive well-substantiated theses.
COTS-based development

The main differences between COTS and traditional development are:

- Development essentially occurs through combining existing products.
- The marketplace exerts strong influence.
- A continuous trade-off happens during development, between requirements, existing products and architecture.
COTS-based development
Methodology

- Systematic literature investigation.
- Structured interviews.
  - 7 small-medium sized companies from Italy and Norway.
  - Asked about their project, and use of COTS products in it.
Definition of COTS

- From interviews: "not produced here", "not modified", "not a commodity", "general purpose".
- From literature: "No control over evolution", "Black box (no source)", "General purpose", "API availability"
- Literature doesn’t focus on the "not produced here" aspect, implied by "commercial"
Definition of COTS

A COTS product is a commercially available or open source piece of software that other software projects can reuse and integrate into their own products

- Isn’t produced exclusively for the project.
- Can be closed or open source.
- Isn’t a commodity.
- Is included in the final product, not a development tool.
- Isn’t controllable
Thesis 1

*Open source software is often used as closed source.*

- Despite code being available, developers don’t look at or modify it, just use product.
- Contrary to popular belief that open and closed source are different.
Integration problems result from lack of standard compliance; architectural mismatches constitute a secondary issue.

- According to literature, architectural mismatches are the primary problem during integration.
- In practice, different versions of, or lacking standards compliance of interaction protocols such as CORBA, COM, EJB, SQL, are a bigger problem.
Thesis 3

*Custom code mainly provides additional functionalities.*

- According to literature, custom code is mostly used for integration.
- In practice, it is often used to add functionality not present in the COTS products.
Developers seldom use formal selection procedures. Familiarity with either the product or generic architecture is the key factor in selection.

- Despite several structured, formal selection processes having been proposed, familiarity is most important in practice.
Thesis 5

Architecture is more important than requirements for product selection.

- According to literature, requirements are most important for selecting COTS product.
- In practice, it is more important to find a product matching the selected architecture.
Thesis 6

Integrators tend to influence the vendor on product evolution whenever possible.

- According to literature, integrators must accept that they cannot control the COTS product.
- In practice, integrators have several ways to do this, and it is common practice, especially for open source.