COTS, INTEGRATION AND CRITICAL SYSTEMS

Victoria Stavridou
Department of Computer Science
Queen Mary and Westfield College
Mile End Road
London E1 4NS
email: victoria@dcs.qmw.ac.uk

1 Introduction

COTS-based software procurement and systems integration are closely related. Considering software development as a process of assembling collections of existing components has potentially several significant advantages:

- Reduced software development costs.
- Reduced software maintenance costs.
- Improved reuse.

This view of systems integration using COTS is attractive, a fact attested to by the US Navy's policy of 1993 that "a government solution and not a COTS solution constitutes disapproving the COTS solution" [7]. The significance attached to COTS in the US is further attested by the initiation of a substantial research programme at the Software Engineering Institute (SEI).

2 Impact on process definition

Traditionally, the capabilities of a software system have been driven by system requirements as shown in figure 1 [1].

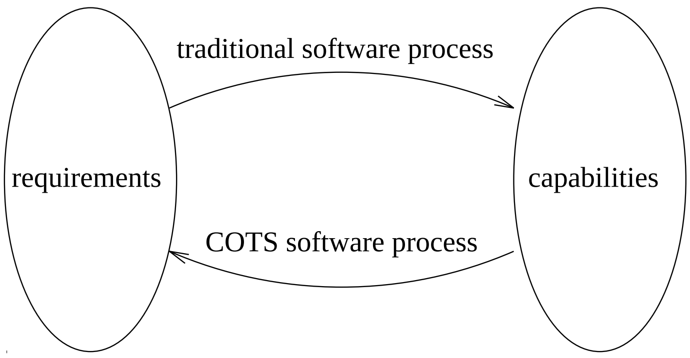

Figure 1: Software process with(out) COTS

A COTS driven software process is fundamentally different since capabilities determine requirements. In this shifted paradigm, requirements that cannot be met using existing capabilities are not possible to define.

Clearly, such a 180 degree shift in the software process is not desirable and it is best to consider a hybrid COTS/bespoke software approach (figure 2). Even so, as Boehm points out, the use of COTS components has significant implications on lifecycle models. For example, the use of COTS has necessitated a revision of the spiral model. A practical COTS acquisition process developed by Loral Federal Systems is shown in figure 3 [10].

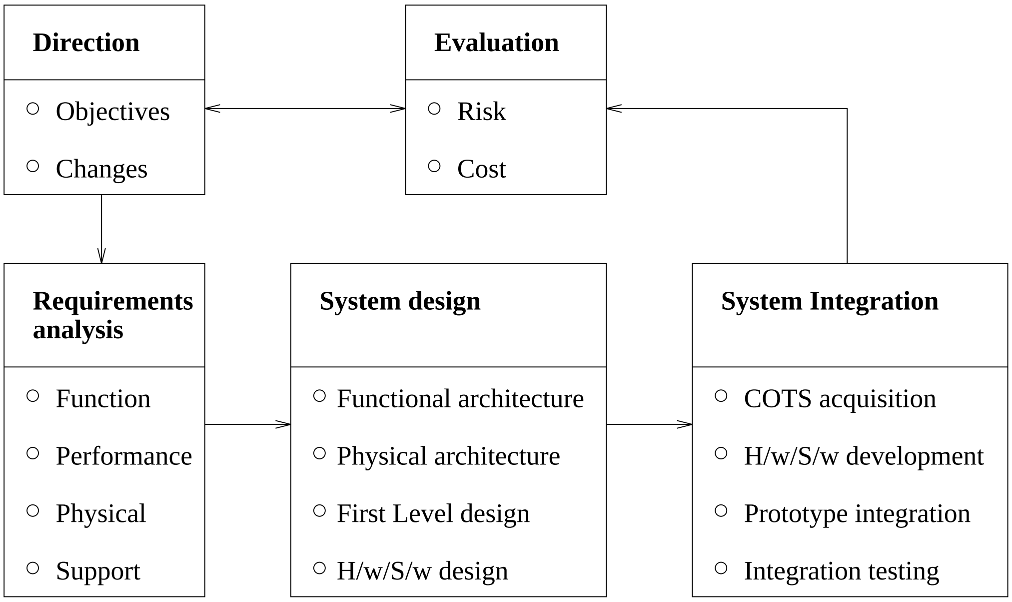

Figure 2: Software process with COTS/bespoke components

3 Impact on integration architectures

Architecture compatibility is another significant issue that needs to be addressed when considering COTS integration. For example, incompatible COTS design assumptions can cause serious interoperability problems affecting data, control, timing and service provision [1]. Presentation integration can also be affected. For instance, using commercial software development frameworks for producing COTS implies that certain compromises must be made on system requirements. So all Macintosh applications look like Macintosh applications as they conform to a certain "look and feel" [8]. If therefore one chooses to use a Macintosh related COTS, the presentation integration of the resulting system must also conform to the same "look and feel".

The way forward from these problems is to address the technology needs that result from the use of COTS which according to Boehm are:

- Mismatch detection, assessment and conciliation.
- Guidance for architecture design, construction and documentation.

Addressing these needs, Garlan [5] suggests that we need to develop:

- Better ways to document architectural assumptions, especially non functional properties.
- First class component connectors such as interaction protocols.
- Architectural analysis tools beyond compilers.

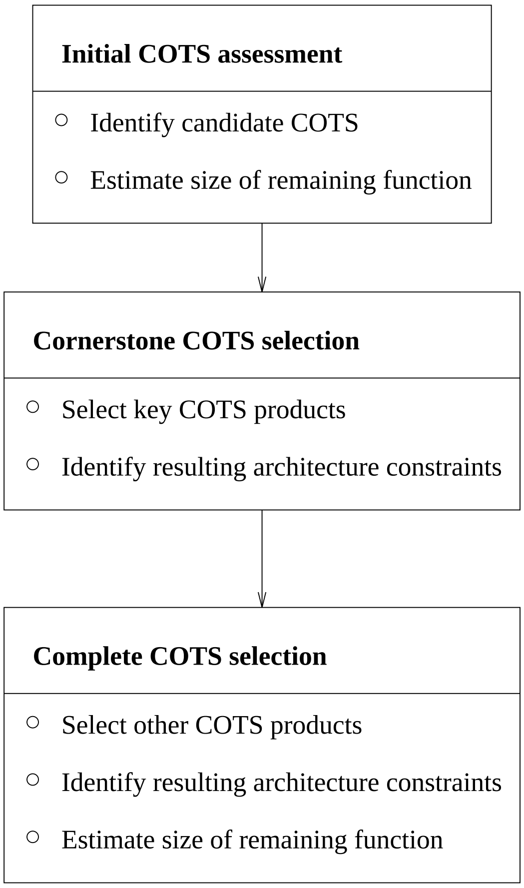

Figure 3: LORAL COTS product selection process

- Culture of orthogonality, e.g. separable parts.
- Better understanding of solution schemes and their domain of applicability such as wrappers, filters etc.

But the cleanest solution of all is the adoption of an open systems approach [4]. COTS software can be most effectively integrated in an open system architecture. Open systems in turn require standards thus underlying the close relationship of COTS to standards.

4 COTS and high integrity systems

The use of COTS in integrated systems that are safety or security related, raises a number of additional issues. This is not to say, however, that COTS cannot be used for such systems. For example, the Boeing 777 has 4 million lines of COTS distributed over 1,000 processors [6]. The use of COTS in this occasion is credited with having reduced development and maintenance costs and improved product portability and enhancement.

Despite such successes, integrating COTS into high integrity systems does raise some significant questions that need to be dealt with:

- *Are COTS suitable for customised, mission-critical systems?* Verification and validation is difficult since the COTS vendor is unlikely to have followed the stringent development practices or standards usually required for such systems (for example, reverse engineering as required by Defstan 00-55 [2] is not feasible unless source code is available which is highly unlikely in the case of COTS). This is understandable since a systems crash is not generally considered critical for the average COTS user. In fact since COTS are primarily aimed at the big volume market, it is difficult to change their development processes to fit the needs of a few big organisations such as the MOD. Boeing used Microsoft products for its 777. To Bill Gates, Boeing was too small a customer to meet with! [6]. The question of certification is very important. Using Microsoft Windows to perform flight-critical roles is not an option contemplated by most experts [9]. A possible solution, shown in figure 4 is to have a hybrid mix, with some software used for critical systems and COTS used for non-critical purposes, with a "firewall", subject to standard certification, protecting the critical software.

Figure 4: Hybrid architecture for high integrity systems incorporating COTS

- *Documentation, acceptance and maintenance* are also issues of concern. The quality of available documentation and training is very important for the process integration attributes of a critical system incorporating COTS. A further question is who is responsible for maintaining the COTS part of the system.
- *Performance* can be critical for many high integrity systems. The performance of COTS components may not be obvious from advertised capabilities. Therefore, vendor demonstrations are not sufficient. Complete product evaluations are required instead.
- *Integration* of COTS into high integrity systems can typically only be achieved by SEI CMM Level 3 organisations [6], presumably because they are able to deploy good systems engineering practice.

5 Summary

The use of COTS in critical systems raises a number of significant issues involving the software process, the choice of system architecture and the applicable standards. The payoffs can be high in terms of improvements in functionality, schedule, lifecycle costs and risk. Inevitably the inappropriate use of COTS can exacerbate problems precisely in these areas. In the case of critical systems certification is a central issue likely to be affected by choosing COTS. By and large existing defence standards preclude the use of COTS because of reverse engineering requirements. A solution in this respect may be the judicious use of service history (as envisaged for example in Defstan 00-56 [3]). A further possibility is the use of firewalls between critical kernels and COTS which effectively means that only the firewall itself needs to be certifies. In this context it would seem profitable to review solutions used in security applications, where for instance the concept of a firewall is well understood and other interesting properties such as non-interference have been extensively studied.

Acknowledgements

This paper and the work herein, would not have been possible without the inspiration and support of Kevin Geary. Parts of the paper have been extracted from an unpublished, internal MOD report.

References

[1] B. Boehm. COTS and Systems Integration: Some Technical Considerations. In A.W. Brown, D.J. Carney, and M.D. McFalls, editors, *Procs of the SEI/MCC Symposium on the Use of COTS in Systems Integration*, SEI Special Report CMU/SEI-95-SR-007, June 1995.

[2] *Draft Defence Standard 00-55 on the Procurement of Safety Critical Software in Defence Equipment.* Ministry of Defence, Directorate of Standardisation, Kentigern House, 65 Brown Street, Glasgow G2 8EX, August 1995.

[3] *Defence Standard 00-56 on Safety Management Requirements for Defence Systems.* Ministry of Defence, Directorate of Standardisation, Kentigern House, 65 Brown Street, Glasgow G2 8EX, 1997.

[4] M.A. Duniho. Towards Open Systems. In A.W. Brown, D.J. Carney, and M.D. McFalls, editors, *Procs of the SEI/MCC Symposium on the Use of COTS in Systems Integration*, SEI Special Report CMU/SEI-95-SR-007, June 1995.

[5] D. Garlan et al. Architectural Mismatch: or why it's hard to build systems out of existing parts. In *Procs of 17th International Conference on Software Engineering*, April 1995.

[6] C.M. Del Fosse. A Commercial/Business Perspective. In A.W. Brown, D.J. Carney, and M.D. McFalls, editors, *Procs of the SEI/MCC Symposium on the Use of COTS in Systems Integration*, SEI Special Report CMU/SEI-95-SR-007, June 1995.

[7] J.G. Hekman. Panel presentation on Department of Defense Needs in COTS and Systems Integration. In A.W. Brown, D.J. Carney, and M.D. McFalls, editors, *Procs of the SEI/MCC Symposium on the Use of COTS in Systems Integration*, SEI Special Report CMU/SEI-95-SR-007, June 1995.

[8] W.L. Scherlis. COTS and Systems Integration: A Technical Perspective. In A.W. Brown, D.J. Carney, and M.D. McFalls, editors, *Procs of the SEI/MCC Symposium on the Use of COTS in Systems Integration*, SEI Special Report CMU/SEI-95-SR-007, June 1995.

[9] J. Stonecash. Towards Open Systems. In A.W. Brown, D.J. Carney, and M.D. McFalls, editors, *Procs of the SEI/MCC Symposium on the Use of COTS in Systems Integration*, SEI Special Report CMU/SEI-95-SR-007, June 1995.

[10] N. Walters. Systems Architecture and COTS Integration. In A.W. Brown, D.J. Carney, and M.D. McFalls, editors, *Procs of the SEI/MCC Symposium on the Use of COTS in Systems Integration*, SEI Special Report CMU/SEI-95-SR-007, June 1995.