A UK view on the problems of logistic support and total cost of ownership estimations for future commercial software based naval combat systems

Authors

[1]Michael Looney
Defence Evaluation Research Agency (DERA), UK
Combat Systems Department
DERA LBTS
Southwick Road
Cosham, Hants, UK
PO6 3RU
Tel: + 44 23 9231 2650
e-mail: mjlooney@dera.gov.uk

Shane Lunga
Defence Evaluation Research Agency (DERA), UK
Combat Systems Department
DERA LBTS
Southwick Road
Cosham, Hants, UK
PO6 3RU
Tel: + 44 23 9231 2771
e-mail: sblunga@dera.gov.uk

[1] Point of Contact

Abstract

There has been a fundamental change to the way the next generation of military combat systems will be procured and developed. Gone are the stove-piped, custom built systems of old when the defence budget was top dog, the new era is demanding a new approach. Mastering through-life support is central to future success in the procurement of naval combat systems that will be based on commercial off-the-shelf products, and cost estimation is our main tool. The work presented in this paper is premised on the realisation that for a system to meet all its acceptance criteria it must be built to do so. Likewise for a system to be costed for effectiveness throughout its life it must be built to be costed. Just as it is very difficult to test a system that was built without testability as one of the prime considerations the work presented here is an attempt to shed light on all the issues that should concern builders of systems with long in-service lives to ensure that their goals are met.

1. Introduction

While it is possible to postulate several technical options for how future military combat systems might be supported during their 30 years or more of in-service life, the final choice will lie with the procurement authorities whose primary concern is that of cost. The move to commercially based software for future systems has introduced factors into the support equation which were never strictly of concern to military systems procurers in the past. These include issues such as rate of change of commercial products, obsolescence and withdrawal of vendor support. Affordability and therefore cost is the decisive factor for systems procurers. This cost manifests itself in many guises that are not always obvious but invariably arise from the need for the systems to evolve or be upgradeable. The costs spans areas such as:

- The cost of initial development and deployment
- The cost of software changes themselves
- The cost of post-integration assessment, evaluation and certification of compliance with mission-critical requirements
- The cost of loss of sea or air time for military personnel
- The cost of licensing and redistribution rights

There is therefore an overwhelming need to careful assess the desirability of frequent system upgrades bearing in mind the cost estimates. To address these realisations the shape of the support curve has changed. Gone is the major half life refit at 10 to 15 years, with perhaps a final add-on at 20 years to see the system through operationally. The demise of that approach also means the disappearance of the cost profile and budgeting process that was well understood. We had a handle on initial development costs; the work at the Software Engineering Institute (SEI) has produced a new development paradigm which allows for the necessary trade-offs etc. and the work at University of Southern California (USC) has produced a cost model, Constructive COTS Software Integration Cost Model (COCOTS), which can be used with that development approach. Through life costs however still remain a fertile ground for research.

In the main part of this paper we will look at some research initiatives that are being pursued to address through life costs. These will inevitably be less formalised but be descriptive enough to aid understanding of potential solutions. In section 2 we briefly present the UK Ministry of Defence (MoD)

’s view of the challenge for system procurers. Section 3 discusses the research initiatives concentrating on the practical steps being taken. Section 4 tries to elaborate on the lack of fundamental theory with which we can formulate hypotheses that we can test.

2. The MoD Procures Naval Combat Systems

2.1 The Problem

The MoD procures naval military systems that include both the vessels and the software systems that go with them. These systems are expected to last about 30 years and are very expensive to develop and keep operationally capable for that length of time. Military budgets are continually falling while the pace of technological change is speeding up.

Every three or so years the MoD has to puts its case to the government for its projected funding requirements. Technological changes and constantly changing mission types make cost prediction extremely difficult. What the MoD essentially needs is a procurement process that will allow it to say to the UK government treasury ‘This is the system we want to acquire and this is how much it will cost to develop, deploy and maintain for the next 30 years’.

2.2 The MoD’s System Development: The View

The MoD assets it can achieve its objective of acquiring systems better and cheaper by following a strategy that has COTS products providing most of the capability of their systems, ‘The adoption of COTS within the Future Attack Submarine (FASM) IT systems is not an issue, it will happen….. The widescale use of COTS will require continued through life and accepted funding for a technology refresh policy….. To be able to make informed decisions on the strategy and approaches that are used, a cost model must be produced for the whole submarine….’ [DERA/SS/WI/WP980106/B.0].

3. Research Initiatives

3.1 Current Practical Initiatives

The necessary research to address the issues associated with the total through life costs of a commercial software based system is still in its infancy. In the UK Ministry of Defence (MoD) there is a major research programme which is looking at de-risking future naval combat systems in terms of facilitating future upgradeability, affordability and capability. In that research initiative there are some basic investigations being carried out into the technical options that are open for through life support.

As a starting point the UK MoD has recognised a few non-technical enabling fundamentals for successful future cost estimations; the identification of the necessary processes; the generation of the appropriate policies and the means to ensure they are followed. In the area of through life cost the approach being adopted is to look at the spectrum of technical options and to identify two or three which appear to have the necessary characteristics in terms of acceptable process and technical capability.

The first stage is the identification of the spread of options and the obvious starting point is with the most market driven which would be to change any software component whenever a new release had been developed and made available. This will require a means of knowing the state of each system in terms of configuration of existing components built into the system and a detailed knowledge of the supply market for each of those components as well as some means of keeping track of new releases. The current state of all systems would then need to be upgraded with the new release in some controlled fashion. However this would not necessarily be a straight forward process because of the possible ‘ripple effect’, - the upward or downward dependencies which can so often be found in commercial packages. It would require investigation of the impact on the other components which might use the new package or be used by it. It will be necessary to identify the process supporting this approach in some detail to ensure that the full implications are understood and captured. This process is then taken as the basis to generate an appropriate cost model extension for COCOTS which can then be populated to provide the information needed to assess the total cost of ownership of the system.

At the other extreme it is necessary to consider the process by which the software can be upgraded at a major refit point during the system in service life. This would be very much in line with the existing support approach but there needs to be some assessment of the implications of adopting this paradigm. The main problem would be in the way in which obsolescence could be handled, i.e. the difficulties of being able to support the system even if the vendor had ceased to support the product. While there might only be a small problem for the first few years this would grow with time and might be unsupportable. The necessary analysis of component change and their implications in both the hardware and software domains must be done to provide the full understanding of the process and its implications before attempting to generate an appropriate extension to the cost model.

Having looked at the two extremes it will then be necessary to identify some intermediate paradigms which could lend themselves to more detailed analysis. This is probable the most difficult aspect of this piece of research. One approach might be to examine the approaches being taken by other major projects in this area such as the US New Attack Submarine or the F22 which has already had problems with chipset obsolescence. Another would be to consider the approaches which have been put forward from the more academic or commercial domain which talk about ‘throw away software’ or, in a similar fashion, the open source model. However, as an initial step in identifying the necessary process options between the two extremes already given, it has been agreed within our research programme to consider a two year upgrade cycle and to use that as the basis for the first attempt at generating a through life cost model. This is in some ways similar to the ‘throw away’ approach in that the system software will be issued at the end of development and an upgraded set of software issued at two yearly intervals to replace the existing configuration.

Our approach is multi-faceted; the first stage is to identify the types of software components and the frequency with which they change. Then, using our experience in the use of COTS software components gained from the research over the last few years, to come up with some level of ‘ripple effect’ which could be applied depending on the type of component being changed. In parallel work is underway to generate some process model which will identify the necessary tasks that will need to be performed to achieve an upgrade cycle and to derive from that some form of cost model which can be applied. There is also work looking at the indirect costs which are associated with the testing and acceptance aspects of any change. While it should be possible to identify the fixed costs of carrying out any laid down acceptance procedure it may well be necessary to reconsider the levels of testing and acceptance which need to be carried out on the final platform. The standing costs per day of using an operational platform will have to be balanced against the ability to achieve an adequate level of testing in some integration environment which could provide the necessary test capability for many aspects of system acceptance. These are issues which will impact the acceptability of any new support paradigm for commercially based military combat systems.

3.2 Lack of reliable cost modelling techniques

Despite the popularity of techniques such as COCOMO and COCOTS the practicalities of software cost estimations that can reliably guide systems procurers in through life budgeting are not well understood. The reason is not surprising if we realise that software will always be a conceptual construct; data sets, functions and their invocations, and relationships between data items[Brooks, 1986]. These remain the same in whatever domain we build software. The problem with software is therefore its flexibility which allows the same constructs to be used in many different domains while what is sought is repeatable patterns that can be recognised in other systems. The tendency with us system designers and ultimately those trying to extract predictable cost models is we inevitably focus on the solutions. We talk of ‘technology insertion’ and ‘technology refresh’ which in essence are solutions to problems. Insertion and refresh are patterns of solutions which we then use for cost modelling. We attempt to model the cost involved in inserting new technology to increase capability or refresh the technology to against obsolescence and since ‘software systems are partial simulations of the problems we become easily persuaded that a description of the software is a description of the problem’[Jackson, 1994]. As a consequence we over-emphasize the solution pattern and use it to make a cost model. The answer to cost estimation may lie in the problem analysis.

Since software solutions reduce ultimately to the same conceptual constructs that span many domains the discriminator between software cost estimations lie in the problems encountered. Developing software for combat systems and that for car engine management systems reduces to data items and their relationships, functions and their invocations etc. The problem domains are however very different. Johnson, an expert in the current hot topic of software patterns, says in [Johnson, 1994]: “We have a tendency to focus on the solution, in large part because it is easier to notice a pattern in the systems that we build than it is to see the pattern in the problems we are solving that lead to patterns in our solutions to them”. We contend that if the MoD could find patterns in the problems that it encounters in through life costs then it could at least start to have a handle on the cost models that could be populated with metrics to give reliable estimations. A cost modelled problem that recurs in a domain is more likely to be unique to that domain than a solution cost model. The recent rise to the fore of component based development testifies to the popularity of solution oriented thinking than problem focussed analysis. This is not to say that the former has no place it has, except that the authors asset that problems would be better guides to the cost models in a particular field. Identification of recurring problems within combat systems through life support should be the starting point to getting the cost models.

4. Summary

The main combat system research programme will provide the means by which the possible paradigm cost models can be exercised in the form of an experimental system test bed which will be used to validate aspects of the support paradigm process model and any cost model developed over an extended period. The research will be carried out with close links into other national and international programmes of research in this area. It is hoped that at the end of the work the UK MoD will be much better placed to formulate its policy in the area of long term logistic support for the next generation of commercially based military combat systems.

At present we lack adequate cost generalisations that would help procurers bring cost estimations under sufficient intellectual control. Due to this absence of a coherent theoretical basis for long term cost modelling most emphasis has been placed on practicalities for initial integration coding effort, which although of value offer limited predictive powers. What we end up doing is collect data or cost metrics without a hypothesis we want to confirm. A combination of problem analysis and case studies of past support strategies may help produce cost models that we can use with confidence. The flexibility of software which allows it to be applied in different fields while using the same basic conceptual constructs suggest that finding cost models maybe better achieved by trying to find patterns not only in solutions but more importantly in the problems.

[Johnson, 1994] Ralph E Johnson; Why a Conference on Pattern Languages? ACM SE Notes, Vol. 19 Number 1, pages 50-52, January 1994.

[DERA/SS/WI/WP980106/B.0] Unpublished DERA report.

[Jackson, 1994] Michael Jackson, Problems, Methods and Specialisation (A Contribution to the Special Issue of SE Journal on Software Engineering in the Year 2001) pp 6-7.

[Brooks, 1986] Frederick P Brooks, No Silver Bullet – Essence and Accidents of Software Engineering Information Processing86: Proceedings of the IFIP 10th World Computer Congress, 1986

References

The views expressed in this document are purely those of the authors and do not reflect the views of the Defence Evaluation Research Agency.

Crown Copyright 2000 - DERA
Published with the permission of DERA on behalf of the Controller HMSO