Exploiting software supply chain business architecture: a research agenda

Barbara Farbey & Anthony Finkelstein
University College London,
Department of Computer Science, Gower Street, London WC1E 6BT, UK
{b.farbey|a.finkelstein}@cs.ucl.ac.uk

0. Abstract

This paper addresses economic and management issues in software engineering. It outlines a research agenda aimed at improving the economic return on software investments, based on a better understanding of the relationships between the costs, benefits and risks associated with system requirements and the way in which the software is designed, developed, acquired and delivered.

1. Introduction

The way in which organisations acquire and develop software is changing and in consequence the economics of systems delivery are changing too. This paper presents an initial analysis of the changes, taking forward a view of software as a product in a marketplace, rather than as an in-house development, and starting an exploration of what that might mean in terms of strategies for acquisition and development. Specifically we want to ask whether there are gains to be had by attending to issues of supply chain management in the context of software and if so, how this could be managed organisationally.

However, before we are able to ask or answer the questions relating to supply chain management with some hope of a practical answer there are a number of other, prior issues which need to be addressed. First, the idea of software provision as the provision of a product or service in a marketplace has to be clarified and some boundaries drawn. For example, large ERP systems may consist of a melange of sub-systems, some commercial off-the-shelf systems, some legacy systems and some integration systems plus whatever adaptations may be developed by individual users. We need to be sure where exactly the product and service metaphors apply and hence where notions deriving from manufacturing or service industries are properly transferable.

Second the idea of a supply chain, or supply network, in software development has also to be clarified and justified. There is, so far as we know, almost no prior work on this subject labeled as such, although there will be overlaps with work on outsourcing.

Third, assuming that we can overcome these first two hurdles, we wish to move quickly to some of the most recent ideas in supply chain management, specifically those coming from Fine's work at MIT (Fine, 1999). The thrust of this work is to suggest that product, process and supply-chain design are best done simultaneously. From our current viewpoint this seems intuitively plausible in the case of software development. But here too there are prior questions that need additional research, specifically about costs and risks: how these are estimated, what is included, how they relate to requirements and how they are shared across the supply chain.

So this paper sets out to do two things: to present the background to the questions and to develop a research agenda, filling in what we know already and presenting the rest as focused research questions. To focus our thinking and to contribute to an ongoing stream of research, we take as our specific target within the software development process, Requirements Management (RM), because:

- the RM process is often weak;
- RM involves both suppliers and customers at each interface in the chain;
- the relationship between costs, benefits, risks and requirements is relatively unexplored;
- relatively small changes to requirements can lead to very large overall benefits.

2. The Software Marketplace

For the greater part of its history software, specifically the complex software systems which will be the focal point of this paper, have been developed in-house or as bespoke systems for individual customers. In this arrangement, two figures dominate the scene: the "user" and the "developer". The reality was always somewhat more complex. Users included user-managers and sponsors for example, and developers had a number of personalities – the systems analyst, programmer, systems designer. Nevertheless the crucial interaction as described in textbooks and papers has focused on just two "camps", systems engineers and users.

But contemporary practice is *very* much more complex. The range of influential participants and stakeholders for any one software system is likely to be very broad. Conversely, as embedded and software shaped systems proliferate in commercial products such as cars and telephones, there is much software for which there is no direct user at all (Stevens et al, 1998). And the organisation of systems development has changed. Outsourcing of large portions of software development is a commonplace.

The implications of these changes are manifold. In this paper we pick up just two:

- In large areas of practice, software is effectively a product and its maintenance a service i.e. there is a diverse market for software and associated services that operates through a variety of forms of contract.

- In organisational terms there is a growing separation between people who build software and people who have to implement it. The gaps are filled by market intermediaries who are largely specialized. The "user" has become a "consumer" (Sawyer S., 1998) and the "developer" fragmented in a supply chain or perhaps more accurately a supply network..

3. Descriptions of the Supply Chain

A supply chain is a series of linked activities which result in a product or service delivered to a final user. In general terms it encompasses ideas like Just-In-Time, Total Quality Management (TQM) and management of customer-supplier relationships.

Of these ideas, it is probably the last, supplier-customer relationships, which will be key to the processes of Requirements Management when viewed through a supply chain lens. This is because effective RE depends on mutual understanding and the emergence of a common language across the customer-supplier interface. It is also because in a marketplace the content and the nature of the contract are critical. The contract will embody the nub of the requirements and, because the contract lies precisely on the customer-supplier interface, it reflects the nature of that relationship.

Moreover the relationship between a requirements engineer and a client calls for a considerable investment in time and goes beyond the formal contract into an informal one. A relationship of this kind is an investment that needs to be considered at a number of levels: financial, personal and/or inter-organisational and strategic. Research suggests that there is a value to the relationship per se which should not be discounted when considering questions of cost, risk and value in a supply chain relationship. This research suggests too that managers need to be aware of the whole network when

making decisions about supply relationships. Strategizing in networks is a series of complex trade-offs with potentially multi-level effects. (Ford et al 1996). Requirements are, we suspect, are one of the important aspects of the trade-offs that are made.

A view of what a supply chain might look like is set out below. Figure 1 shows the generic supply chain. The Table which follows shows the major variants that need to be taken into account. Note how in each case costs and risks shift between different parties.

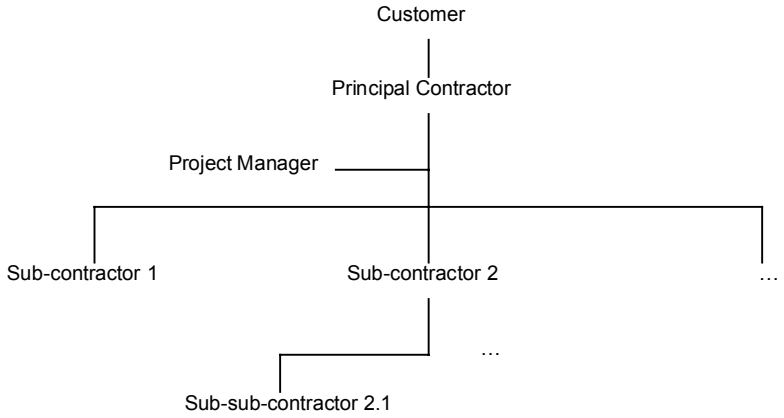

Figure 1: Generic supply chain

Principal Contractor	Internal cost centre
Principal Contractor	Internal profit centre
Principal Contractor	Outsourced
Customer	Requirements role outsourced
Subcontractors	Within customer framework agreement
Subcontractors	COTS vendors included in supply chain
Subcontractors	Legacy system developers & maintainers included in supply chain

Table 1: Significant variants of supply chain architecture

This is a rough characterisation, we need to find out exactly what the patterns are in practice and this requires analysis of existing cases in the literature, backed up if necessary by new case work

4. Contemporary Supply Chain Research

New research at MIT (MIT, 1998; Fine, 1999) suggests that significant increases in value can be obtained through simultaneous design of product, process and the supply chain. Their contention is that, although product and process design are developed together in the most advanced companies, supply chain design is *ad hoc*, made up as they go along. Fine argues that in choosing which skills to retain and which to outsource, equivalently the "make/ buy decision", or in our terms the design of the supply chain it is "the skills that support the very process of choosing" that are crucial (Fine 1996). He further positions the skills required as essentially systems engineering skills, arguing that product development is a repeated sequence of systems engineering, decomposition and outsourcing decisions and gives some tentative guidelines as to the criteria for outsourcing.

If software is a product then software development, including requirements engineering, should benefit from the conscious application of this approach. Intuitively, the increases in value will come from a) the ability to recognise and utilise the expertise of the specialist suppliers in meeting system

requirements, b) improved delineation of the limits and possibilities of the requirements space early on in development and c) a better understanding, earlier on, of the trade-offs that are available between fulfilling a requirement and the cost of implementing it. This is just a guess, unsubstantiated by evidence. We need a programme of research to assess it.

5. Research Agenda

The fundamental question of the research agenda is:

In software-dominated systems is there scope for:

- increased competitive advantage;
- a reduction in time to market;
- lower costs;
- reduced risks;
- better value.

through concurrent design of the product, the software process and how the software is to be supplied, in short the "software supply chain" architecture. The term "architecture" is used in the sense of John Kay's work on Corporate Strategy (1993). Kay defines "architecture" as "a network of relational contracts within, or around the firm". So the "software supply chain architecture" is the network of relational contracts, internal or external, surrounding the supply of software. It is not the software system architecture.

The overall aim of the research would be to provide practical recommendations for reducing time to market, costs and increasing value added in the case of market driven software intensive systems, by improving the design and management of the supply chain. Specifically the aims would be to effect improvements to the requirements engineering process through concurrent design of requirements, requirements process and software supply and, since this is an essentially practical project of research, to suggest how they may be managed organisationally.

The starting point is the MIT work. However, as we have said, in the particular case of software engineering, there are some related questions to be answered first.

Given these gaps in what we know, it is likely that trying to answer the fundamental question in one go will be too difficult. Instead we see four stages in the research agenda:

- **Stage 1.** Underpinning of the ideas of a supply chain and supply chain architecture in the context of software engineering. Examination of current options and practice

- **Stage 2.** Holding requirements and requirements process fixed, as it were, assessment of the costs benefits and risks associated with different supply chain architectures

- **Stage 3.** Holding the form of supply chain architecture fixed, as it might be say under a framework agreement or strategic alliance, assessment of the differential costs, benefits and risks associated with different sets of requirements

- **Stage 4.** Assessment of the impact of designing the requirements, requirements process and supply chain architecture together.

Each of these gives rise to more detailed research questions:

Stage 1.

- What is a "software supply chain"?
- What constitutes a supply chain "architecture"
- What architectures do we find in practice?
- Is there anything to be designed or are the parameters in practice more or less fixed?

Stage 2

- Given a set of requirements and different possibilities for the form of a supply chain, what are the potential costs, risks and benefits associated with each form, including the investment in the relationship?
- How are these costs, risks and benefits currently distributed in practice in each case?
- Given a set of requirements what kind of improvement in costs, benefits and risks can be expected overall with an "optimised " supply chain?
- Who would gain and who stands to lose?

Stage 3

- For each supply chain architecture, how would a change in requirements impact the pattern of costs, benefits and risks?
- How are sets of requirements moulded by the prior existence of a software supply architecture?
- What is the balance between the advantages of a "fixed" architecture, for example being able to take advantage of the domain knowledge of a long-term outsourcer, and its disadvantages, for example being locked in to that outsourcer.

Stage 4.

- Is it possible, technically and organisationally to design the requirements, process and architecture together?
- What gains are there to be had?
- How will the gains (losses) be appropriated and distributed?
- What are the organisational implications?

We see a number of different types of study as being necessary to support the research agenda. The basic platform would be a series of case studies and critical reviews of relevant literature. The later stages require the development of models, for example of cost and risk functions, and theories which support "what if" scenario analyses.

An immediate gap surrounds the questions relating to value. There are very few studies of the economics of software acquisition alternatives. A major requirement for the research agenda would be to provide, via case study work, at least ballpark figures on the measurable aspects of cost, benefit and risk associated with a set of requirements and a supply chain architecture.

What is vital at each stage is to extract the organisational lessons from all of the case-studies, theory and scenarios and condense these into usable lessons for practical application.

6. Summary

The agenda outlined above is considerable. It requires a range of skills not least the quantitative skills for developing economic models and the qualitative skills required for in-depth, insightful case study work. Our conclusion is that the agenda is not for a project but perhaps a programme. It is a roadmap, setting out what is required and how it might be attempted and, as part of the research, to establish the economic and organisational benefits to be had.

References

Fine C.H. (1999) *Clockspeed: winning industry control in an age of temporary advantage* Chaps. 5 and 11, Perseus Books, Reading, Massachussetts

Fine C.H. and Whitney, D.E. (1996) *Is the make-buy decision a core competence*?, MIT Center for Technology, Policy and Industrial Development, February, 1996

Ford D., McDowell R. and Tomkins C. (1996) Relationship Strategy, Investments, and Decision-Making, in Iacobucci D. (ed.) *Networks in Marketing*, Sage, London

Kay J. (1993) *Foundations of Corporate Success: how business strategies add value,* Oxford University press, Oxford, Chapter 5

MIT (1998) Creating and Managing Corporate Technology Supply Chains: Value Chain Design in the Age of Temporary Advantage, The MIT Series on Technology and the Corporation, May 12-13, 1998, Cambridge, Massachusetts

Sawyer, S. (1998) Seminar, London School of Economics and Political Science, London

Stevens R., Brook P., Jackson, M., and Arnold S., (1998) Systems Engineering, Coping With Complexity, Prentice Hall Europe, Hemel Hempstead