Component-Based Business
Background Material

On Determining the Requirements for Software Components

Richard Veryard, October 2000

Key Questions

How will software engineering practices need to change, if they are to accommodate Component-Based Development?

How soon do you expect these changes to be widespread within the software industry?

Critique of Traditional Software Methods

This analysis in this paper enables us to identify the limitations of traditional software methods, such as OOAD (object-oriented analysis and design).

Focus on user requirements

Traditional engineering methods, including OOAD, start from the assumption that somebody **requires** something to use, and the engineering task is to construct a **solution** that satisfies that user requirement.

This assumption fits most software development projects, but not all. Some of the time the project objective is to build something that nobody knew they needed. The only known requirement may be the supplier's need to remain in business, but this is not usually regarded as a user requirement, and is usually ignored by requirements engineering methods.

But if CBD projects are supposed to be focused on reuse, this conflicts with the supposed need to be focused on user requirements. This is because **reuse has no meaning within the service use ecosystem**.

Focus on software system behaviour

Furthermore, most traditional engineering methods formulate the requirements in terms of the behaviour of some system, usually restricted to a computer information system or software system. OOAD methods usually specify these requirements as a set of **use cases**.

Where a software development project can exercise design control over an entire system, then this approach still seems to be valid. However, this approach fails to support the design of components within open distributed systems, because such systems generally lack central design control.

> Is it possible to define a use case for a single component? Do you think it is meaningful or useful to do so? Or would you regard this as a trivialization or perversion of the use case concept?

Furthermore, there are problems with flexibility. The demand for flexibility (which we have positioned within the device use ecosystem) needs to be balanced against the demand for particular services (which we have positioned within the service use ecosystem).

> Is it possible to formulate requirements for flexibility in terms of use cases?

Methodological Implications of Our Analysis

Any CBD method that is truly focused on software reuse and economies of scale should contain the following elements.

- A systematic way of understanding the design and operational constraints of a given ecosystem.
- A systematic way of defining component interfaces that will be useful, meaningful, relevant, and compatible within the target ecosystem.
- A systematic way of predicting the amount of use that a given component is likely to achieve in a given ecosystem.
- A systematic way of balancing standardization with customization.

Such a method will almost certainly need to articulate several different viewpoints on a distributed system.

Possible Approach

Viewpoints

We define five viewpoints.

Viewpoint	Focus	Key Concepts
Enterprise	Business relationships	Stakeholder, Intention, Role, Responsibility
Exchange	Conversations Workflows and information flows	Joint action, Collaboration, Transaction, Flow
Behaviour	Activities	Service, Interface, Rule, Operation, Use cases
Design	Components	Component, Component kit, Connector
Physical	Infrastructure	Platform, Mechanism, Protocol

Table 1 Five viewpoints – generalized

Not all five viewpoints may be applicable or relevant in every situation or project.

Scenarios

This method allows for several scenarios of user/requirements, including the traditional ones:

- Provision of whole systems or individual components for a specified user or user community, funded by (or on behalf of) that user/community.
- Speculative development of whole systems or individual components, for publication and sale within a defined market.
- Development of intelligent software agents for release into a global network (such as the Internet), with various payment and funding models.
- Collaborative speculative development, in which a relatively small number of users participate in a development, but a significant portion of the funding comes from speculation against future reuse by a much wider community.

Steps

We can analyse the requirements for software components according to the steps outlined in Table 2

Scope Ecosystem	Define the target ecosystem for components.	We use informal techniques to produce an initial scope definition, which may be a combination of text and rich pictures. Later refinements are more formal, and are based on the models of the ecosystem.
Model Ecosystem	Develop formal descriptions of the ecosystem.	We may produce models in all five viewpoints, or some of them, depending on the situation.
Identify Service Opportunities	Identify unfilled or weakly filled niches in the service use ecosystem.	A service opportunity may be based on any of the following: An unsatisfied intention of some agent An unsatisfactory existing service, lacking quality or availability An opportunity to subdivide existing roles.
Create Service Opportunities	Specify interface of proposed service.	Must include quality of service characteristics as well as 'functional' behaviour.
Confirm Service Meaningful within Ecosystem	Estimate the likely adoption of the proposed component in the target ecosystem.	We check the specification against our analysis of the ecosystem, in all five viewpoints. In some cases, the use of a component depends on achieving a critical mass within the ecosystem within a defined period.
Confirm Service Economies	Verify that the service can be delivered cost-effectively and profitably.	This may be based on an analysis of the opportunities to reuse existing assets, or to otherwise leverage economies of scale.
Extend Ecosystem	Identify additional ecosystems for potential use.	If the proposed component is not cost-effective or profitable within the original target ecosystem, or it is unlikely to achieve a critical mass of usage, we may need to find additional ecosystems to support the development of this component.

Table 2 Steps towards a methodical approach

Further Reading

Albert Borgmann, ***Technology and the Character of Contemporary Life.*** Chicago University Press, 1984.

Francisco Varela, ***Principles of Biological Autonomy.*** New York: Elsevier Science Publishers, 1979.

Richard Veryard, ***Plug and Play: Towards the Component-Based Business.*** Springer-Verlag, 2000 (forthcoming).

Richard Veryard, "Reasoning about Systems and their Properties". To be published in Peter Henderson (ed), ***Systems Engineering for Business Process Change Volume 2***, Springer-Verlag, 2001.