Pratical Software Reuse, Michel Ezran, Maurizio Morisio and Colin Tully, Springer-Verlag London Limited 2002

Page 2

Software reuse is the systematic practice of developing software from a stock of building blocks, so that similarities in requirements and/or architecture betweeen applications can be exploited to achieve substantial benefits in productivity, quality and business performance.

Page 5

Assets are work products of any kind, from any part of the software process. ‘Asset’is an appropriate word, since software work products capture knowledge that is importn to the enterprise, and therefore carry potential value. Reuse is a powerful means of exploiting that value-adding potential.

Page 6

The definition refers to ‘a stock of building blocks’. If that stock comprises more than a small number of items, we need to make arrangements to now what items we have, where to find them and whether they are worth keeping; otherwise we will not derive the best return from having established the stock. In this respect, a stock of reusable software assets is no different from any other kind of stock. A smal shopkeeper can see or remember the contents of his store-room; the big store needs techniques of stock control and management. Managing a stock of reusable software assets, described… uses concepts such as the asset catalogue and the asset repository.

Having an effective catalogue is often essential is achieving systematic reuyse. Its absence can be major contributor to chaos. It is rather like the situation, familiar no doubt to many parents, where Lego blocks are scattered all over the house, in every room, so that you don’t know which ones you’ve got, the ones you need you can’t find, and the ones you find by chance you probably don’t need.

If a catalogue lists what assets we have and where they are stored, then a repository (like a library or a warehouse) is where we store and retrieve them. Reusabel software assets may be stored in a single repository, or there may be multiple repositoroes: that is like the difference between keeping all Lego blocks in a single big box, or keeping the members of different sets in their separate boxes.

As can be easily imagined, how to design repositories for reusable software is of great interest to those of a technical inclination, and there is much discussion of it in the technical literature. Should we use database technology, or a configuration management system, or the repository of an integrated software engineering environment? Should the catalogue and repostory functions be combined or separate? How should assets be classified for storage and retrieval, and how should search requirements be specified?

Page 27

Reusable assets are composed of a collection of related software work products that may be reused from one applicatino to another. In this book, the word ‘component’ is used with a specific meaning: a software component is an executable asset that may be integrated as-is into an application. A component is a specific type of aset that refers to component-based technologies.

Page 30

Assets may have very different sizes and granularities. For instance: a function or a procedure; a class; a group of classes; a subsystem, a framework or a service; an application or a product.

Assets are often nested (an asset contains several assets). The reuser is free to reuse the overal asset or one of its nested assets.

The biffer the assset is, the earlier it should be taken into account in the application development process.

A reusabe asset is potentially made of manu life-cycle products: requirement definition; architecture definition; analysis model; design models and code; test programs; test scenarios; test reports.

A single asset is made of a set of related work products. These work products can represent the same piece of software at different abstraction levels (requirement, a..), and thus each work-product of the asset can be reused at teach step of the life-cycle (before and after coding).

The reusable material must also be packaged with all the information necessary to reuse it (asset ‘meta-infomration’, or asset description (Karlsson, 1995)): classificatino infomration to facilitate rapid retrieval of suitable assets; description to facilitate understanding of what the asset does; documentation to facilitate undesrtanding of how to use it (and if necessary to customize it); qualification and test information to facilitate evaluating and testing it; information about its origin to facilitate obtaining support or additional information.

Page 49

The first question to be asked is probably: does a company invariable need a central repository and, if not, what are the conditions in which a repository is required? A company or a development team can afford not to have a defined and managed repository for assets provided that the developers are able, even in its asence, to know what assets exist and where they are. This situation is not normal and could be summed up as one where both the number of people and the number of assets are very limited. In general, the factors to be considered in setting up a repository are: the number of developers, development teams and development sites in the company; the number of assets, and the number of different domains to which the assets belong;

As these numbers grow, assets have to be managed in an organized and defined way. An essential part of this management effor is to define and maintain a catalogue of assets. This will avoid the situation in which developers do not know about (a) an existing asset that would match part of their need, (b) where to look for this asset or who owns it, and (c) what to do with ta new potentially reusable asset.

A repository to store assets offers the following advantages: the definitions and common recognition of a place for assets, therefore a known and unique place to look for and deposit assets; a homogeneous way of documenting, searching and accounting for assets; a defined way of managing changes and enhancements to assets, including configuration management procedures.

…

In a few words, a reuse repository should be able to identify assets, associate a description to each of them, and maintain links to the work products that contribute to each of them, and maintain links to the work products that contribute to its body. In detail, a repository could offer the following functions.

Asset identificatino and description. Asset insertion. Catalogue Browsing. Textual search. Asset retrieval. Organization and search (keyword, faceted classificationm thesaurus). History. Measurement. Access Control. Version management. Change control. Change notification.

…

Page 59

CMS tools define a place where work products at the project level are stored, so that project members can access them on a disciplined basis. CMS tools are usually raw environments, without any functions to define asset descriptions or to organize and search for them. Moreover, they are usually oriented to managing fine-grained elements on a project basis, while reusable assets can be large-grained, containing composite elements and of a cross-project nature both in space and time. Of course CMS tools are very suitable as regards access control, change control and version management.

Some missing functions could be defined using the API for the tool, given it exists and it is open. This option should be considered if a company/department has a CMS in place, and no inteface is available to a reuse repository.