

# Guidelines for Reuse in DSDM

## Table of Contents

<b>1</b>	<b>INTRODUCTION.....</b>	<b>2</b>
1.1	AIM .....	2
1.2	AUDIENCE .....	2
1.3	CONTRIBUTORS .....	2
<b>2</b>	<b>BENEFITS OF REUSE .....</b>	<b>3</b>
<b>3</b>	<b>PRINCIPLES OF DSDM REUSE .....</b>	<b>4</b>
<b>4</b>	<b>WHAT TO REUSE AND WHEN.....</b>	<b>5</b>
4.1	CANDIDATES FOR REUSE .....	5
4.2	DECIDING WHEN TO REUSE .....	5
4.3	REUSE IN THE DSDM PROCESS.....	5
<b>5</b>	<b>CHARACTERISTICS OF REUSE ENVIRONMENTS.....</b>	<b>7</b>
<b>6</b>	<b>OTHER CONSIDERATIONS .....</b>	<b>8</b>

## **1 Introduction**

This White Paper discusses issues which are relevant for DSDM developers reusing existing components. It considers benefits of reuse, principles for reuse, reuse in the DSDM process framework, the reuse environment, and various issues.

The extent to which substantial reuse is feasible is largely determined by the technical environment and whether there is a corporate infrastructure to support reuse. At the time a DSDM project is started, the project typically has little control over whether there is such an infrastructure. In this sense reuse is optional in DSDM.

There are various definitions of software reuse. A suitable definition (from Karlsson) is:

*Software reuse is the process of creating (parts of) software systems from existing software assets, rather than building software systems from scratch.*

Another useful definition is that reuse reduces (or controls) redundancy in the delivered system, and in the products and practices used to develop it. There is an important distinction between planned reuse (i.e. reusing components which were designed to be reused), and software salvage. Note that although (re)using existing development tools, methodologies or standards is good practice, we exclude it from our definition of reuse.

Reuse can take place at many levels: within a project/application, across projects/applications within an organisation, or using externally supplied materials. Components may be reused without change, or may be specialised or modified in order to reuse them.

### **1.1 Aim**

The aim of this White Paper is to highlight where reuse can be used to advantage in improving the speed and accuracy of delivering systems within a DSDM environment.

### **1.2 Audience**

The readership is expected to be IT staff who understand some of the issues of reuse and who are conversant with DSDM.

### **1.3 Contributors**

This White Paper is a reissue with some additions of the chapter on Reuse put together for DSDM Version 2 by a Task Group chaired by David Redmond-Pyle. Thanks are due to all contributors to that chapter.

## **2 Benefits of Reuse**

The key benefits of reuse in a DSDM project are that:

- systems can be delivered faster
- higher development productivity can be achieved (through savings in design, coding and testing)
- higher quality systems can be developed (higher levels of usability, maintainability, reliability, or other non-functional requirements can be achieved with available resources)
- less effort will be necessary in testing during development and in maintenance
- greater consistency in styles and standards
- the range of project skills required may be reduced because some areas of complexity are handled by the reused materials.

Note: Although some reuse is possible with most software technologies, the technology which has most potential for extensive reuse is object-oriented technology. The Object Management Group has produced several architectural standards which facilitate development of reusable object-oriented software.

### **3 Principles of DSDM Reuse**

#### **Reuse is a means to achieving the project's objectives.**

In DSDM, reuse is not an end in itself. If an appropriate reuse environment and materials are in place, reuse will typically be adopted as the most effective way of doing the job in hand. If they are not, the DSDM project should be free to use other means to meet its objectives, unless it is explicitly constrained to reuse, e.g. by a standard architecture specified in the project terms of reference.

#### **DSDM projects should not carry the costs of enabling reuse.**

Reuse requires a strategic infrastructure that should be centrally organised and funded, so that DSDM projects do not carry the costs of enabling reuse. There are significant costs in creating and supporting reusable materials. Many of these costs of reuse, such as the cost of a reuse manager, a technical environment and the cost of generalising components to make them reusable, should be regarded as corporate investments in infrastructure and assets, rather than costs attributable to an individual application development project.

#### **In order to reuse, the DSDM team must have knowledge and understanding of the reusable components available to them.**

Lack of developer understanding of, and confidence in, reusable components is frequently a major inhibitor to reuse. One solution to this knowledge problem is to facilitate reuse by secondment of a reuse broker from the reuse group to the DSDM team. For externally-sourced materials, there may be internal or external experts.

## **4 What to reuse and when**

### **4.1 Candidates for Reuse**

Many types of materials can be reused, including:

- Design models (e.g. data models, object models, process models)
- Data dictionaries (e.g. data item definitions)
- Object-oriented application frameworks and class libraries (e.g. Smalltalk library)
- Office systems software integrated into hybrid solution (e.g. wordprocessing, workflow, email, presentation graphics, FAX)
- Software objects designed for reuse within some infrastructure (e.g. OLE servers, OMG CORBA objects, OpenDoc)
- Component libraries (e.g. device drivers, .VBX, .OCX)
- Server-based services (e.g. existing transactions invoked through a client/server interface)
- Module libraries (e.g. NAG library of statistical functions, COBOL common modules)
- User Interface Style Guide elements (e.g. standard controls or images)
- Test scripts and data
- Documentation framework of templates, etc.

### **4.2 Deciding when to reuse**

DSDM projects may work within the context of corporate policy, culture and standards which give guidance on what components to reuse.

During the project the extent of reuse is normally at the discretion of the project team. In this situation, the basis for choice is whether or not reuse helps the team to meet its objectives in delivering functionality to the business, i.e. in weighing the costs and benefits of reuse for this specific project.

A project will be at risk if an inappropriate choice is made of materials to reuse. The costs of reuse may escalate and outweigh the benefits. Another risk is that the DSDM team may start building components for future reuse, incurring development costs not related to the DSDM project objectives. This should be actively discouraged. However identification of components for later development should be encouraged.

### **4.3 Reuse in the DSDM Process**

This section highlights the actions required to exploit reuse in the first four phases of DSDM. It is irrelevant during the implementation phase.

### **Feasibility Study**

Consider whether significant reuse is feasible for the project. Ask what extent can existing models or implemented components are reused. Assess how this may affect project feasibility, cost, schedule and risk. Plan the project's approach to reuse.

### **Business Study**

Check whether any existing models (or parts of them) can satisfy some of the business needs of the DSDM project. Wherever possible, DSDM models should use and extend corporate data models or object models, or the models of related areas produced by earlier projects.

### **Functional Model Iteration**

Consider reuse of application frameworks, class libraries, component libraries or various kinds of tactical reuse, i.e. use of components to quickly and cheaply achieve an effect you want to explore. If usability is addressed in the functional prototype, questions of application style guides and reusable components that support or enforce them are relevant.

### **Design and Build Iteration**

Exploit and extend application frameworks, class libraries, component libraries. Design new components to be interoperable with reused components. Test new and specialised functionality with reusable components to implement issues that were not addressed during the Functional Model Iteration.

## **5 Characteristics of reuse environments**

A DSDM project is most likely to exploit reuse effectively if there is an environment which facilitates reuse. There are several aspects to this environment.

### **Cultural**

Management and developer attitudes to reusing code as opposed to writing new code are important, e.g. the “Not Invented Here” syndrome. If reuse receives lower peer valuation or lower financial reward (e.g. productivity bonuses are adversely affected) it is unlikely to be widely adopted. Any metrics to measure programmer or project productivity should reflect the value of reuse.

### **Organisational**

Reuse is unlikely to occur without an organisation unit that has responsibility and a budget for promoting reuse by making it easier and more attractive for project teams, e.g. by establishing technical infrastructure, developing reusable components, advertising available components and advising on their use. Where reusable components are owned centrally, there needs to be a communication channel to notify all parties concerned about any change to a reusable component. The optimum structure for all of this is a reuse centre or asset management group with secondment of reuse brokers to project teams.

### **Technical infrastructure**

The infrastructure should include some or all of the following.

- technical aids to reuse include repositories of components
- tools to search for and retrieve components and browse code
- impact analysis tools and configuration management specifically for reusable components
- links to external suppliers of components
- data dictionaries
- CASE tools.

## **6 Other Considerations**

### **Costs of reusing**

Reuse has costs as well as benefits. Project costs include:

- learning about reusable materials
- effort to adapt or specialise them
- costs caused by constraints or limitations of reusable materials
- possible additional testing complexity.

Specialisation and testing can involve developers having to spend time understanding complex structures which are irrelevant to the current project requirements.

### **Ownership/contractual issues**

When reusable components are included in a system, issues may arise over the ownership of parts of the delivered system, or over ongoing rights or obligations in relation to them e.g. licence fees, responsibility for defects, and maintenance. These should be clarified prior to reuse.

### **Evolution of reused components**

Reusable components typically have their own development cycle outside the DSDM project. New releases may become available, either to correct defects or add enhanced features, and issues arise about whether, when and how to upgrade to newer versions of components.

### **Reuse and Configuration Management**

CM has sometimes been seen as primarily archival, as making sure that you can retrieve and reconstitute a particular software configuration to rebuild a release. The reuse management and configuration management schemes should be fully integrated. Making a CM environment suitable for reuse involves issues such as visibility of classification schemes and other semantic relationships between components, and a sufficiently usable user interface for it to be suitable for DSDM developers on an hour-to-hour basis.