



## SOFTWARE REUSE \*

### 0 EXECUTIVE SUMMARY

A literature search has been carried out for information about the application of software reuse in military avionics and other defence or embedded systems. The information found, including over 50 published papers and a number of World Wide Web pages, is described under three headings. The first heading considers reports of the application of software reuse to the production of real systems. The second heading describes a number of initiatives aimed at promoting software reuse or collecting libraries of reusable software. The final heading collects papers which consider the progress of software reuse and some of the issues raised by attempts to adopt reuse more widely.

The search has found a large amount of relevant information, describing extensive activities being undertaken to make use of software reuse in avionics and related systems. Almost all the information is from the USA, relating either to projects carried out by US defence suppliers for the DoD or resulting from the DoD's Software Reuse Initiative. The vision statement of the DoD's Software Reuse Initiative indicates that the DoD has adopted a policy to promote software reuse.

It is not clear whether the relative lack of information from UK and other European countries is a result of a lack of activity or of a reluctance to release information about work in progress. There is no indication, however, that any European effort to adopt reuse has achieved the level of activity being undertaken in the USA.

---

\* **This report is published by the Avionic Systems Standardisation Committee (ASSC) to advance the role of standardisation in avionics. The use of this is entirely voluntary, and its applicability and suitability for any particular use, including any patent infringement arising therefrom is the sole responsibility of the user.** Copies of this paper are obtainable from the ASSC Agency as an Avionic Systems Standardisation Committee publication.

Blank page

<b>Table of contents</b>		<b>Page no.</b>
<b>0</b>	<b>Executive summary</b>	<b>1</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>5</b>
	1.1 Task Background	5
	1.2 Objectives	5
	1.3 Approach	5
	1.4 Structure of the Report	5
<b>2</b>	<b>Overview of types of Software reuse</b>	<b>5</b>
	2.1 Reuse of Requirements, Design or Source Code	5
	2.2 Personnel and Organisation	6
	2.3 Reuse with or without Modification	6
	2.4 Planned or Opportunistic Reuse	6
	2.5 Reuse Component Size	6
	2.6 Software Component Libraries	7
	2.7 Object-Oriented Languages	7
	2.8 Software Reuse Economics	7
<b>3</b>	<b>Examples of Software Reuse</b>	<b>7</b>
	3.1 Military Avionics	7
	3.2 Other Military Systems	8
	3.3 Civil Avionics	10
	3.4 Company Reuse Programmes	10
	3.5 Success Stories from the ReuseIC	11
<b>4</b>	<b>Software Re-Use Initiatives</b>	<b>12</b>
	4.1 The Comprehensive Approach to Reusable Defense Software (CARDS) Program	12
	4.2 Portable, Reusable, Integrated Software Modules (PRISM) Program	13
	4.3 National Software Data and Information Repository (NSDIR)	13
	4.4 Software Technology for Adaptable, Reliable Systems (STARS) Program	14
	4.5 Defense Software Repository System (DSRS)	14
	4.6 Reuse Information Clearinghouse (ReuseIC)	15
	4.7 The Army Reuse Center (ARC)	15

<b>Table of contents (cont)</b>	<b>Page no.</b>
4.8 Reuse Library Interoperability Group (RIG)	16
4.9 Asset Source for Software Engineering Technology (ASSET)	16
4.10 USAF Reusable Ada Avionics Software Packages (RAASP)	17
4.11 Common Ada Missile Packages (CAMP)	17
4.12 Reuse Based on Object-Oriented Techniques (REBOOT)	18
4.13 Avionics Domain Application Generation Environment (ADAGE)	19
4.14 Software Reuse Initiative (SRI)	19
<b>5 Assessing the Progress and Future of ReUse</b>	<b>20</b>
5.1 Assessing the Achievements of Reuse	20
5.2 Management and Business Issues for Reuse	21
5.3 Technical Issues for Reuse	22
5.4 Future Directions	22
5.5 DoD Policy	23
<b>6 Conclusions and Recommendations</b>	<b>23</b>
6.1 Conclusions	23
6.2 Recommendations	24
<b>7 References</b>	<b>25</b>

## **1 INTRODUCTION**

### **1.1 Task Background**

The work described in this report has been carried out for the Processing Working Group (WG) of the Avionics Systems Standardisation Committee. The issue of software reuse arose following a discussion of software obsolescence at the 5th Processing WG meeting held on 25 January 1996.

### **1.2 Objectives**

The objective of the work is to investigate current work on software reuse by:

- 1) examining any successful examples of developments making use of software reuse, especially in military and related applications, such as embedded real-time systems
- 2) identifying any software re-use initiatives relevant to avionics, such as software component libraries.

### **1.3 Approach**

A literature search was carried out for references relating to software reuse in avionics. Over fifty references were considered to be relevant. A small proportion of these articles have been obtained, while for others only the abstract was examined. On-line news articles were also searched for further information: these have provided background information but they are not explicitly referenced. Various techniques were used to search the World Wide Web (WWW), including Web Crawler and Alta Vista. This search found a large amount of information, primarily relating to activities resulting from the DoD's Software Reuse Initiative (SRI).

### **1.4 Structure of the Report**

In Section 2 an overview of software reuse is given, including criteria which can be used to evaluate examples of software reuse. Papers which report on software reuse examples are described in Section 3. Information relating to software reuse initiatives is presented in Section 4, identifying the organisation responsible for the initiative and describing the activities being carried out. In Section 5, overviews are given of papers which discuss the progress of software reuse and issues which are raised by the adoption of software reuse, including commercial and legal issues. Conclusions are in Section 6.

## **2 OVERVIEW OF TYPES OF SOFTWARE REUSE**

Any evaluation of reports of software reuse needs to take account of the variety of practices which can be described as reuse. In this section, the characteristics of different forms of reuse are described.

### **2.1 Reuse of Requirements, Design or Source Code**

Although software reuse normally refers to reuse of source code, it is also possible to reuse design, or other products of the software lifecycle, such as requirements. If a

software development project has to comply with the requirements of a particular software development standard, it may not be possible to reuse source code without also reusing or recreating a description of design.

## **2.2 Personnel and Organisation**

Reuse occurs within an organisation context. If the personnel involved in software reuse were also the original developers, they are likely to have knowledge which would not be available if the same reuse was attempted by other personnel. Reuse of software outside the developing organisation poses further problems, including those of IPR ownership and liability for errors in the reused software.

## **2.3 Reuse with or without Modification**

Software can be reused without modification only if it meets the precise requirements of the reusing project. In cases where the requirements are similar, but not exactly the same, some modification is required. Modifying reusable software increases the need to understand the software behaviour fully; in particular, the design of the software as well as its function must be understood.

## **2.4 Planned or Opportunistic Reuse**

Opportunistic reuse occurs when the developers of a system reuse parts of an earlier system, even though this reuse was not anticipated by the developers of that earlier system. This form of reuse occurs in many software development organisations, but it may not be tightly controlled. When software reuse is planned, the software may need to be developed to requirements which are generalised, beyond the needs of a particular application. The analysis of requirements common to a class of applications is called domain analysis; techniques for domain analysis are considered by most commentators to be immature.

## **2.5 Reuse Component Size**

The size of the component of software which is reused varies. Traditionally, most software reuse has been of very small or very large software components. A subroutine from a library of mathematical functions is an example of small scale software reuse. This form of reuse is appropriate only for software components with very simple interfaces; although it is widely practised, it is applicable to only a narrow range of functionality.

Use of a standard database package as part of a system is an example of large scale reuse. This form of reuse is also widely practised, but it lacks flexibility: for example, features of a database which are needed in the application cannot be separated from those which are not required. More recent work on software reuse attempts to overcome the limitations of these forms of reuse, allowing reuse of software components of a size appropriate to each application.

## **2.6 Software Component Libraries**

A software component library contains reusable software components, often for a particular application domain. Some examples are described in Section 4. Setting up a software component library raises several complex problems, for which solutions are only beginning to emerge. A particular problem is how to classify and store the components, so that a user can search the library for a software component satisfying his requirement.

## **2.7 Object-Oriented Languages**

The use of an object-oriented language, such as C++ or Ada95, is not a prerequisite for software reuse. For example, some of the software component libraries described in Section 4 contain COBOL components. Nevertheless, software reuse can be easier in languages which support modules with well-defined interfaces. Object-oriented languages based on classes are strongly modular and also support inheritance, which may allow a software component to be customised without modifications to the source code.

## **2.8 Software Reuse Economics**

Advocates of software reuse stress the possible cost savings which can be achieved by reuse. When the aims of a development project are expanded to include the production of reusable software or when a software component library is established, costs are incurred which have to be set against the savings achieved by reuse.

# **3 EXAMPLES OF SOFTWARE REUSE**

In this section a brief review of the references describing experiences of software reuse is given. The references are grouped in several sections, starting with those relating to military avionics, the area most directly relevant to the ASSC. Sections on other military systems and civil avionics follow. A number of companies have reported on their efforts to establish company reuse programmes and these reports are introduced in the fourth section. A final section mentions some of the 'reuse success stories' catalogued by the Reuse Information Clearinghouse.

## **3.1 Military Avionics**

In [Ref 1, Shelbourne91] an evaluation is given of the Common Ada Missile Package (CAMP) carried out at Wittenberg University. CAMP is described in Section 4.11. The authors conclude that at the time CAMP was not suitable for avionics applications. Recommendations for improvements to allow wider scale application of reusable software are given in the paper.

In [Ref 2, Batory95] a description of the Avionics Domain Application Generator Environment (ADAGE) system, developed as part of the Domain Specific Software Architectures (DSSA) project, is given. The paper describes the reference architecture for avionics systems, which allows avionics software systems to be synthesised from software components using the ADAGE tool. Further details and sources for this project are given in Section 4.13. Since this paper is written by DSSA project members, it does not make an independent evaluation of the ADAGE system.

In [Ref 3, Rosskopf91] an Ada package providing reusable input and output functions for a MIL-STD-1553B interface is described.

In [Ref 4, Gingerich89] the method used to develop software for a mission processor consisting of several types of 1750A-based modules is described. All the modules required communications and Built-In-Test (BIT) software, with common requirement between the systems. A common software library was implemented, achieving 65% reuse for the start-up and BIT software, with advantages such as faster development, less testing and common interfaces.

In [Ref 5, Coglianesi92] the creation of a workstation-based environment to support the development, maintenance and upgrade of avionics systems with an order-of-magnitude improvement in quality and productivity over current approaches is discussed. The author proposes to achieve this improvement through the reuse of large portions of well-designed and documented software. The challenges of capturing and representing knowledge relating to avionics applications, such as navigation, guidance and flight control are described to illustrate the impact of designing a domain-specific software architecture. The technologies required to enable engineers to use domain knowledge effectively within a workstation-based environment are discussed also.

### **3.2 Other Military Systems**

In [Ref 6, Svoboda94] the progress made and the lessons learnt by the Application Engineering Team of the demonstration project carried by Unisys for the US Army under the STARS programme is described. STARS is described in Section 4.4.

The demonstration project aimed to achieve both re-engineering and reuse, using a combination of domain engineering and application engineering. The domain engineering team carried out domain modelling and the creation of software assets for reuse. The application engineering team was responsible for re-engineering and maintenance of existing systems and the development of new systems. The paper contains a description of the application engineering process developed to address issues relating to systematic reuse, emphasising the benefits of co-operation between the application and domain engineering activities.

In [Ref 7, Smith94] the experience of the Fire Support Software Engineering (FSSE) organisation, which is part of the US Army Software Engineering Directorate is given. FSSE have instituted a formal software reuse program, which requires common standards, procedures and practices to be applied universally within each application domain. The paper describes the past methods and future trends, including the practical experiences of introducing a reuse program.

In [Ref 8, Barlin92] the development of a real time embedded message processing system, the Submarine Message Buffer (SMB) is described. The paper focuses on the design processes and methodology used for the SMB, which depended on the reuse of

Ada software. Metrics are included. Other work on reuse of Ada within the US Navy undersea application domain is described in [Ref 9, Juttelstad92].

[Ref 10, Ruegsegger88] contains an early assessment of the RAPID software library concept. RAPID, the Reusable Ada Packages for Information System Development, was developed by the US Army Information System Engineering Command, which develops US Army information systems. A US Army personnel system serves as a testbed for RAPID and related issues of software reusability. RAPID is also described in [Ref 11, Reugsegger90]. This paper draws on the experience of the RAPID program to determine the requirements of an effective software reuse policy.

In [Ref 12, Henry95] the results of two large industrial projects carried out by Matra Cap Systemes are described. These projects used both project-based and cross-organisational software reuse to improve both productivity and quality. All the systems are command, control, communication and information (C<sup>3</sup>I) systems, operating in different environments. System A, the original system, was developed for the Navy for use offshore; it comprises 400K lines of C++. Although no reuse of this software had been planned, the same project team achieved 37% and 35% improvements in productivity and fault rates by reuse of 35% of System A for the development of a second system, System B. System B is a shore based C<sup>3</sup>I system, used by top Navy commanders and comprising 470K lines of C++.

System C is a mobile vehicle-based C<sup>3</sup>I system, for use by Army commanders. A total of 34% of code from System B was used in a fully-operational prototype of System C. The Proto System C development was carried out in a different organisation, and involved only three out of 27 members of the System B development. Although the overall reuse figure is similar to that achieved for System B, the reuse only applied to two out of seven functional sets; in one of these 76% of the code was reused.

The reuse strategy followed by Matra Cap was based on pragmatic, opportunistic reuse. The pragmatic reuse strategy was considered to offer the best prospects of short-term pay back, since it does not involve the same level of investment as a strategy based on planned reuse. As part of the reuse initiative, the company amended its software development procedures to allow for reuse, created a company wide database of reusable assets, using Web technology and carried out an education programme. One problem created by the success of the initiative is the creation of multiple versions of similar modules. To minimise the maintenance costs, it is now proposed to maintain reused modules centrally. Some activities have also been carried out to prepare for a programme of planned reuse, including the preparation of a business case, some domain analysis projects and the establishment of criteria for evaluating reuse opportunities and assets.

### **3.3 Civil Avionics**

In [Ref 13, Kirby95] the challenges faced by aerospace avionics systems developers in the 1990's are described: namely developing reusable designs, meeting all customer needs on-time and within budget. The author takes the Boeing 777 electrical load management

system (ELMS) as an example of a large system development and describes the advantages of those process approaches and types of tool which worked, and the disadvantages of those approaches which were less successful.

Further details of software reuse for the Boeing 777 project are given in the ReuseIC (see Section 4.6) WWW page at <http://sw-eng.falls-church.va.us/reuseic/pubs/flyers/boe-reus.htm>.

NASA [Ref 14, McGarry92] has been conducting experiments, for 16 years, in software engineering technology toward a goal of understanding and improving the software process used on NASA projects. The author discusses some of the major lessons that have been derived.

### **3.4 Company Reuse Programmes**

A number of companies have reported the results of reuse programmes or other experiences with reuse.

In [Ref 15, Krutz91] a critical appraisal of the benefits of reuse is given. The experience of Intermetrics, a software company who have incorporated reusable software components into an organised reusable software library, are described, noting the costs of this process and the problems encountered.

Eight years of experience of software reuse at AT&T is reported in [Ref 16, Belanger94]. The software discussed in the paper includes well known tools used both within and outside the company. A high degree of reuse has been achieved: the paper records the lessons learnt about the configuration and distribution processes required.

Three recommendations for implementing software reuse are given in [Ref 17, Joos94] arising from a programme carried out at Motorola. The importance of support from top management and training of engineers are emphasised, with reuse incentives and reuse tools being recommended also.

Hewlett-Packard, which had divisional reuse efforts in place since the mid-1980s, initiated a corporate reuse programme in 1990. As described in [Ref 18, Griss94], the first objective was to gather information about the existing reuse. Better methods for domain-specific, reuse based software engineering were then developed. It is concluded that the obstacles to successful reuse are mostly non-technical. The influence of organisational factors on the success of reuse in Hewlett-Packard are described further in [Ref 19, Fafchamps94]. The paper describes four patterns of interaction between the producers and consumers of reusable software, which were found to exist in Hewlett-Packard, and considers the advantages and disadvantages. There is no recommended pattern; instead the importance of understanding the culture of an organisation and the strains that a reuse programme may impose on the culture is stressed.

A further report on software reuse in Hewlett-Packard is given in [Ref 20, Lim94]. The article presents metrics from two HP reuse programmes, evaluating the quality and productivity increases achieved. As an example, one of the programmes reported a 76% reduction in the defect-density of reused and generated code, compared to new code. The article also presents an analysis of the overall economic benefits of the two software reuse programs, over 10 and 8 years since their inception. This analysis offsets the savings achieved by reuse against the start-up and recurring costs of reuse, to calculate the net present value and the break-even point. One of the projects reached break-even after two years, while the other required six years.

Different aspects of reuse are described in two papers from the IBM Systems Journal. IBM has established a Reuse Technology Support Center to co-ordinate and manage reuse within IBM. [Ref 21, Bauer93] describes the experience gained during the evolution of the center; the importance of language support for reuse and of high-quality class libraries are stressed. In [Ref 22, Tirso93] management aspects of the program are considered. Despite some mixed results, the program has expanded to encompass 30 IBM sites and has delivered benefits including cost savings, improved performance or functionality, and reduced maintenance.

In [Ref 23, Rosenbaum95], a software reuse effort carried out by Schlumberger Oilfield Services since 1988 is described. A Common Software Library group has been built up, with responsibility for managing 17 software component libraries containing 2 million lines of software for use in oil data acquisition products. The importance of people management issues to the success of reuse is emphasised and a model for the introduction of a software reuse program is offered.

An experiment into software reuse for avionics carried out by British Aerospace in 1988 is described in [Ref 24, Hutchison88]. The experiment analysed the potential reuseability of different software components in a particular application domain, with or without modification.

### **3.5 Success Stories from the ReuseIC**

The Reuse Information Clearinghouse (ReuseIC) has documented a number of reuse success stories, available on the WWW (see Section 4.6). Short reports from the US Army, Bofors, Fujitsu, GTE Data Services, Magnavox, the SH-60R Helicopter Upgrade Program, NASA, NEC, Raytheon, SofTech, Toshiba and Universal Defense Systems are included. A few of these, most likely to be of relevance to the ASSC, are described below.

Five systems from the US Army's Tactical Command and Control System (ATCCS) were found to have commonalties. Reuse is estimated to have achieved cost avoidance of \$479M.

42% of software required for the SH-60R upgrade is being reused. Sources of the reused software include the P-3 Update IV program, an Advanced Electronic Support Measures program carried by Loral, avionics developed for the SH-60B and other GFE.

Universal Defense Systems, which develops command and control applications, has a library of 396 Ada modules totalling 500K lines. With this base, they have developed the Australian Maritime Intelligent Support Terminal with 60% reuse.

#### **4 SOFTWARE RE-USE INITIATIVES**

In this section, a number of software reuse initiatives are described. These initiatives include the production of libraries of reusable software components, trials of software reuse techniques in development projects and the distribution of information on software reuse.

##### **4.1 The Comprehensive Approach to Reusable Defense Software (CARDS) Program**

**Organisation** US Air Force sponsored program

##### **Description**

The CARDS program is dedicated to furthering DoD and Government agency objectives of widespread introduction of systematic software reuse into software acquisition, development and maintenance. The CARDS mission is to develop a knowledge base of reuse products and processes and to perform technology transfer to other government organisations.

The CARDS program is structured around the key elements of the DoD Software Reuse Vision and Strategy. At the highest level, the DoD vision for reuse is to drive the DoD software community from its current method of developing software from scratch. The DoD's aim is to construct software components that are held in libraries based on a specific architecture and belonging to an application domain. Within this vision, planned reuse becomes an essential facet of each phase of the software development life cycle.

CARDS advocates the product-line approach supported by a systematic reuse-based systems engineering discipline. A product-line is a collection or family of software systems with similar or overlapping functionality and a common architecture that satisfies the mission requirements for that set of systems. A product-line approach involves the development and application of reusable assets (technology base), the development of systems from a common architecture, and large scale reuse of high quality assets.

**Contact** E-mail: [hotline@cards.com](mailto:hotline@cards.com)  
Tel: 1-800-828-8161

**WWW Location** <http://dealer.cards.com/CARDS/>

## 4.2 Portable, Reusable, Integrated Software Modules (PRISM) Program

**Organisation** Part of the CARDS program, sponsored by the US Air Force.

### Description

The PRISM program applies hands-on experience and pragmatic systems engineering to minimise risk and reduce the development time and costs associated with command centre acquisitions. Operational command centres developed using PRISM concepts will be standards-based, allowing reusable components to be integrated faster, cheaper and with fewer errors as compared to traditional software development approaches. By applying PRISM concepts, it is claimed that the potential exists to reduce software acquisitions by 3 to 4 years and hundreds of millions of dollars.

All qualified PRISM products and/or documentation are placed into the CARDS Command Center Library for reference and future use by command center acquisition organisations and contractors.

### Contacts

Mr Robert Lencewicz - Program Manager

Tel : (617) 377-9369 or E-mail: lencewicrz@tango-vs1.hanscom.af.mil

Lt Alfonso Rosabal - GCC Program Manager

Tel: (617) 377-4285 or E-mail: rosabala@tango-vs1.hanscom.af.mil

Mr Andrew Hodyke - Deputy Program Manager

Tel: (617) 377-8473 or E-mail: hodykea@tango-vs1.hanscom.af.mil

**WWW Location** <http://www.cards.com/PRISM/>

## 4.3 National Software Data and Information Repository (NSDIR)

**Organisation** US Air Force sponsored program

### Description

NSDIR is a managed store of data (and resulting information) about software development and maintenance projects. The repository collects, manages and analyses the results of software engineering processes and products. NSDIR is chartered to benchmark technologies and methodologies utilised in Air Force software-intensive projects through the collection and management of software product and process metrics data. Although initially focused on USAF projects, the NSDIR is tasked with obtaining similar metrics data from other sources beyond the USAF.

### Contact

NSDIR, ATTN: User Support  
Lockheed Martin Tactical Defense Systems  
1000 Technology Drive, Suite 3130

Fairmont, WV 26554  
 Tel: (304) 366-1175 Fax: (304) 366-2382

Problem Reporting/Help Request E-mail: nsdir-help@cards.com  
 Submissions E-mail: nsdir-submit@cards.com

**WWW Location** <http://www.cards.com/NSDIR>

#### **4.4 Software Technology for Adaptable, Reliable Systems (STARS) Program**

##### **Organisation**

The STARS Program is sponsored by the Defense Advanced Research Projects Agency (DARPA). Prime contractors are Boeing, Loral Defense Systems-East and Loral Federal Systems.

##### **Description**

The STARS program is focused on changing the way software is developed within the DoD. The change envisaged is a shift to a product-line concept for software development and life-cycle support, characterised by an architecture-based approach to software engineering within application domains.

This paradigm shift supports the STARS goal to increase software productivity, reliability and quality and has been accomplished by integrating support for modern software engineering processes and reuse concepts in software engineering environments (SEE).

There are three STARS demonstration projects:

1. Army/Loral Defense Systems-East STARS Demonstration Project
2. Navy/Boeing STARS Demonstration Project
3. Air Force/Loral Federal Systems STARS Demonstration Project.

The STARS program is described in [Ref 25, Anderson85].

**Contact** E-mail: [info@stars.ballston.unisysgsg.com](mailto:info@stars.ballston.unisysgsg.com)

**WWW Location** <http://source.asset.com/stars>  
<http://www.stars.reston.unisysgsg.com/stars/loraldse/>

#### **4.5 Defense Software Repository System (DSRS)**

##### **Organisation**

Under the management of the Center for Software Reuse Operations (CSRO) on behalf of the DoD.

**Description**

The DSRS is an automated library of reusable software development components available to the DoD. It serves as a central collection point for quality assets, and facilitates software reuse by offering developers the opportunity to match their requirements with existing software products. Although primarily used to support reuse and object-oriented Ada development products, the DSRS will support storage and retrieval of products other than Ada related products. Written in Ada, the DSRS uses the Oracle database management system and operates on the UNIX (SunOS) platform.

**Contact** E-mail: perryj@cc.ims.disa.mil

**WWW Location of DSRS Catalogue**

<http://www.ssc.af.mil/EN/ENSD/AFRC/Catalog/catalog-intro.html>

**4.6 Reuse Information Clearinghouse (ReuseIC)**

**Organisation** Sponsored by DoD's SRI

**Description**

The ReuseIC serves as the DoD's Software Reuse Initiative's (SRI) information dissemination centre. Gives access to information on software reuse news and events, source code repositories, publications, lessons learned, successes, methodologies, working-groups, policy and history and links to other reuse internet resources.

**Contact** Reuse Information Clearinghouse (ReuseIC)  
P.O. Box 1068  
Falls Church, VA 22041  
Tel: 703/681/2471, Fax: 703/681/2868  
E-mail: reuseic@sw-eng.falls-church.va.us

**WWW Location** <http://sw-eng.falls-church.va.us/>

**4.7 The Army Reuse Center (ARC)**

**Organisation** US Army, Department of Defense

**Description**

The ARC was established as a directorate of the US Army Information Systems Software Center (USAISSC) on 1 January 1994. Its mission is to develop, implement, maintain and administer a total reuse program supporting the entire software development life cycle. Its mission will help the Army fulfil the DoD objective to introduce reusable, maintainable and reliable software. It was established to support the development and deployment of reliable, high quality systems, while reducing the time and resources required to develop and maintain those systems.

**Contact**      Army Reuse Center  
 USAISSC  
 ATTN: ISQB-IRC Stop C2  
 Fort Belvoir, VA 22060-5576

FAX: (703) 806-3864  
 Customer Assistance:      (703) 806-4300  
 Library Assistance:      (703) 806-4283

DSRS Admin:                  arcopns@issc.belvoir.army.mil  
 Newsletter:                  arcnews@issc.belvoir.army.mil  
 Customer Support:      lloyd@issc.belvoir.army.mil  
 Technical Support:      laformed@issc.belvoir.army.mil

**WWW Location**      [http://arc\\_www.belvoir.army.mil/](http://arc_www.belvoir.army.mil/)

#### **4.8      Reuse Library Interoperability Group (RIG)**

##### **Organisation**

Membership consists of various US DoD agencies, NASA, IBM, Military electronic suppliers (such as Raytheon Company), and Telecommunications companies (such as MCI Telecommunications).

##### **Description**

Currently there are no commonly practised methods for sharing assets among the many government and industry software reuse libraries. RIG claim that inefficiency and unnecessary redundancy results. Instead it is proposed that Interoperability, the sharing of assets among reuse libraries, can be the key to improving efficiency and increasing the value and impact of reuse libraries in the short term and avoiding an explosion of incompatible protocols in the long term.

The primary purpose of the RIG is to draft standards for interoperability of software reuse libraries and to submit them to formal standards organisations, such as ANSI and IEEE. The Basic Interoperability Data Model (BIDM) has been approved by RIG and submitted to IEEE as a proposed standard.

**WWW Location**      <http://www.rig.org/>

#### **4.9      Asset Source for Software Engineering Technology (ASSET)**

**Organisation** ASSET had its origins in DARPA's STARS programme.

##### **Description**

ASSET's origins were as an on-line software reuse repository in DARPA's STARS

program. As the STARS program comes to an end, ASSET has moved to become self-sufficient through on-line brokering of software components available through their Worldwide Software Resources Discovery (WSRD), supplemented by new selections of commercial software, product catalogues, documents and information sources. ASSET offers products and services in four principal business areas:

the WSRD now contains over 700 certified software assets that can be purchased through their electronic commerce system; ASSET also features on-line interoperability with four other major digital libraries

ASSET's Library offers information related to software reuse, software newsletters, various professional product catalogues and featured articles on prominent individuals in the software community

ASSET also features software brokerage services for individuals or companies wanting to offer their products for sale on the World Wide Web; their growing selection includes products as diverse as software reuse surveys and topical papers offered by economic development organisations and trade associations

ASSET will also design and implement custom digital libraries.

**WWW Location**     <http://source.asset.com/>

#### **4.10 USAF Reusable Ada Avionics Software Packages (RAASP)**

Little information has been identified on this initiative but RAASP is mentioned in the following extract, concerning RIG (see Section 4.8):

"The idea to form the RIG grew out of interaction initiated by Jack Kramer, STARS Program Manager, and representatives of the RAASP library".

In [Ref 26, Beser92], the goals of the RAASP program are stated to be to achieve avionics reuse at both requirements and source code level. The project aims to develop a flexible library of reuse software for use in avionics systems written in Ada.

RAASP is also evaluated in the following paper from the US Defense Technical Information Center: Raghava G. Gowda, 'A framework for developing and managing reusable avionics software', Defense Technical Information Center, DITC #AD-A276-846, September, 1993.

#### **4.11 Common Ada Missile Packages (CAMP)**

**Organisation** US Air Force

##### **Description**

The CAMP products include CAMP parts, the CAMP Avionics Benchmarks, and the CAMP Parts Engineering System (PES). Camp Parts are 444 reusable Ada components

organised into 35 Top-Level Computer Software Components (TLCSCs) which contain 137,000 source lines of Ada code. The CAMP Armonics Benchmarks are used to evaluate Ada and processor implementations in the armonics domain. The tests establish the correctness of compiler implementations and measure performance in size and speed of generated code. The CAMP PES provides mechanisms for identifying and retrieving reusable software parts, adding new parts to the catalogue and data administrator functions. The PES runs on VAX VMS systems. The CAMP products are distributed on magnetic tape. Distribution to any organisation other than those within the US Air Force requires a completed Statement of Terms and Conditions.

The achievements of the CAMP program up to 1991 are described in [Ref 27, Mullins91].

**WWW Location**     <http://www.utica.kaman.com:8001/tools/camp.html>

#### **4.12 Reuse Based on Object-Oriented Techniques (REBOOT)**

##### **Organisation**

Sema Group sae and Bull S.A. have been identified as participants in the project. Although the source of funding has not been identified it is possible that the project is funded by the European Framework research program.

##### **Description**

The REBOOT project studies, develops, evaluates and disseminates advanced methodologies for reuse-driven and object-oriented software development with the emphasis on planned reuse. REBOOT tackles the following problem areas:

- methods for identifying suitable reusable information
- methodology for developing reusable components
- techniques and tools for creating components and their associated descriptions
- repository for components
- techniques and tools for qualifying, classifying, storing, browsing, retrieving, evaluating, displaying and extracting components
- methodology for reusing components
- non-technical aspects.

##### **Contact**

Jean-Marc Morel, Bull S.A., Rue Jean Jaurès, F-78340 Les Clayes-Sous-Bois,  
FRANCE

Tel: +33 1 3080 7448, Fax: +33 1 3080 3379, E-mail: J.M.Morel@frcl.bull.fr

Gabriel Sánchez Gutiérrez, Sema Group sae, Albarracin 25, 28037 Madrid,  
SPAIN

Tel: +34 1 327 2828; ext 204, Fax: +34 1 754 3252, E-mail: gsg@sema.es



only a sample of the large literature on software reuse techniques, which is not the focus of this report. The following section gives an overview of three papers addressing the future of software reuse. The final section covers the vision and strategy of the DoD Software Reuse Initiative.

### **5.1 Assessing the Achievements of Reuse**

There is a wide literature assessing the progress and achievements of software reuse and debating its role. In this section, a sample of these papers is referenced.

In [Ref 28, Coomer90], the importance of developing libraries of reusable software components for DoD systems is argued. Writing in 1990, the authors argue that the existing approaches are not working. The paper focuses on the technical issues, such as the characterisation of reusable components, and presents solutions.

In [Ref 29, Frakes91] a panel discussion held at a conference in May 1991 is recorded. The discussion considered the evidence that software reuse is delivering, as demonstrated by verifiable increases in software productivity and quality.

In [Ref 30, Moore92], an employee of IBM Federal Sector Co assesses the evolution of software reuse in the defence sector. The achievements to date - typically savings of 10% - are reviewed and the particular challenges facing wider adoption of reuse in defence applications are considered. Three developments which are considered to be encouraging are domain analysis, concurrent life cycles and an infrastructure for a national reuse industry.

In [Ref 31, Neighbors94], it is argued that a factor of ten improvement in productivity could be achieved. The author adds that maintaining flexibility in software development to respond to changing requirements is also important.

Writing in 1994 the author of [Ref 32, Tracz94] revisits nine 'myths' about software reuse originally proposed in 1988. One of these myths is that 'software reuse is a technical problem', which has been denied by numerous authors. In [Ref 33, Tracz94], the same author reviews the sources of the most significant advances in software reuse and those areas which have produced less benefits than anticipated. Facts and myths about software reuse are also reviewed in [Ref 34, Basili94]. Noting that there has been much misunderstanding about software reuse, this author picks out three particular misunderstandings, starting with the over emphasis on reuse of code.

The business case for software reuse is considered in [Ref 35, Poulin93]. Since software reuse is not free, the costs of identifying reusable software and integrating it into a new development must be compared with the benefits. Using metrics produced by IBM which reflect the effort saved by reuse, the overall business case for reuse is examined.

## 5.2 Management and Business Issues for Reuse

In [Ref 36, Morrison90], the Strategic Defense Initiative (SDI) program, which is planning a trusted library of reusable software, is discussed. Key issues associated with these technologies will be the subject of investigation by the Strategic Defense Systems (SDS) Software Center. The goal of the program is to achieve lower costs, reduced risk, better reliability, better security and intellectual control over the software development process.

In [Ref 37, Moore94], how a reuse marketplace might be built upon the existence of multiple libraries is examined. The operations of a reuse marketplace are described and possible specialisations of libraries are suggested. The pioneering contributions of some current efforts are noted also.

Conference article [Ref 38, Cardow89] reviews a number of the issues that will hamper the widespread introduction of software reuse for the Air Force. Recommendations are provided on the steps to be taken now to lay the groundwork for reuse to become an accepted reality. It is noted that technology barriers that have prevented reuse from becoming standard procedure are being addressed, but other major barriers like acquisition, technical and managerial issues need to be considered.

The research into the legal issues relating to software reuse carried out as part of the CARDS Program (see Section 4.1) is described in [Ref 39, Huber94]. The legal issues depend on the role of each organisation; this paper focuses on the roles involved in a software component library, including component provider and component user. CARDS has developed an approach to software reuse libraries which aims to minimise the business and legal risks. The paper is intended to motivate discussion among potential users or contributors to software component libraries.

In [Ref 40, Gowda94], the critical issues in managing software projects are examined, and management guidelines for introducing software engineering methodologies and CASE tools within the organisation are offered. These guidelines are offered through a model project which enforces standards for new projects.

[Ref 41, Tirso91] describes a support structure which is used to implement a software reuse program. The focus of the structure is to eliminate the non-technical inhibitors typically associated with reusing software. The structure presented revolves around a site-wide reuse co-ordinator who acts as a reuse champion, reuse focal points within each project, a review board that reviews reusable candidates and appropriate measurements and incentives.

## 5.3 Technical Issues for Reuse

[Ref 42, Jahanian94] presents a re-engineering approach based on the formal specification of a system through decomposition into a collection of services with well-defined interfaces. An approach based on formal specification complements the development of a

toolkit of common services that can be reused in re-engineering multiple embedded systems.

[Ref 43, Benjamin93] explains that the US Air Force (USAF) requires the use of the Ada in the development of new weapon system software. Each Ada compilation system uses a run-time system (RTS) for executive services such as tasking, memory management and system initialisation. However, implementing and using custom RTS services in each software development activity inhibits avionics software reuse and portability.

[Ref 44, Poulin94] debates the issues on whether a centrally-managed reusable software library (RSL) or a domain-specific RSL provides the best focus for a reuse program.

[Ref 45, Skoglund93] discusses ways to improve morale and productivity. It is suggested that this can be done through the implementation of reusable components and suggests that C++ and Ada, suited for embedded systems, contain features that facilitate reusability.

[Ref 46, Shaw95] surveys common architectural styles, including important packaging and interaction distinctions, and proposes an approach to the problem of reconciling architectural mismatches. [Ref 47, Garland95] describes the problems and pitfalls of trying to combine off-the-shelf software, caused by incompatibilities in the architectures of the different components.

[Ref 48, Zand94] discusses the differing levels at which software reuse could be implemented, including the specification level, the design level, program/subprogram library level, code level and object code level, and gives the distinct definition for reuse at each of the levels. It also discusses the techniques applied to organise and manage reuse from one level to another.

#### **5.4 Future Directions**

The major Commission-sponsored programmes represent the main thrust of reuse-oriented activity in Europe. [Ref 49, Favaro94] explains how the focus of that work is high-level and oriented towards holistic, "meta" technology with little impact on the commercial market. The author suggests that this focus must change for Europe to become competitive in future markets for software components.

[Ref 50, Frakes94] speculates on where software reuse is headed. The author concludes that systematic reuse will play a key role in the software industry in the next ten years and beyond and that the companies that do the best job of engineering a domain will have a tremendous advantage over their competition and are likely to dominate markets based on that domain.

[Ref 51, Meyer91] is a general discussion of the economics of reusing software programs or modules. The authors examine the possibility of reusing software in a changed context

and present formulae to determine the cost/benefit ratios for reused software, allowing for relative productivity.

## **5.5 DoD Policy**

The DoD policy on software reuse has arisen from the DoD Software Master Plan. In [Ref 52, Potter90], presented in 1990, the findings of a working group charged to develop a Software Master Plan are described. The working group examined past studies as input for the plan; one of the software issues identified was software reuse.

The DoD Software Reuse Initiative (see Section 4.14) has produced a vision and strategy document, now in its 9th edition [Ref 53, DoD96]. The executive summary of this DoD document starts as follows:

*The Department of Defense has evidence that software reuse principles, when integrated into acquisition practices and software engineering processes, provide a basis of dramatic improvement in the way software intensive systems are developed and supported over their life-cycle. This document describes the vision and strategy for a DoD initiative that will make a reuse-based paradigm the preferred alternative for developing and supporting software. The strategy applies to all types of software-intensive systems managed by the Department.*

The DoD strategy is based on systematic reuse. The strategy consists of the following nine elements:

- 1) identify domain where reuse techniques can be applied
- 2) develop guidelines to identify reusable assets
- 3) establish criteria for deciding ownership of reusable assets
- 4) integrate reuse into the system development life-cycle
- 5) modify the current acquisition process to foster reuse
- 6) develop incentives for the practice of reuse
- 7) define a process to evaluate the benefits of reuse
- 8) invest in technology supporting reuse products and processes
- 9) develop a training plan to ease the transition to reuse.

## **6 CONCLUSIONS AND RECOMMENDATIONS**

### **6.1 Conclusions**

The information contained in referenced papers and available on WWW pages allows a number of conclusions to be reached.

- 1) Many US defence and electronic equipment suppliers are evaluating the use of software reuse. In contrast with the predicted 'Not invented here' syndrome, most of the experience reports are at least partly positive; however, they suggest that at least some companies are making a success of software reuse.
- 2) The DoD has launched a Software Reuse Initiative (SRI) with the aim of making software reuse an integral part of the DoD procurement strategy. As part of the DoD SRI, DAPRA funded research programs have resulted in a number of reuse initiatives, and have created the first software component libraries for military embedded systems.
- 3) Only a small number of references have been found describing activities in Europe, including the UK. The lack of information could be explained by either by a lack of activity or by a less open culture, including less use of the World Wide Web. It seems most likely that both factors are at work.
- 4) Software reuse raises a number of technical and business issues, including ownership, legal liability and the structure of a reuse market place. There is evidence that these issues are being considered, but to date, they appear to be unresolved.

## **6.2 Recommendations**

The results of this information search indicate that the potential for benefits to both the MOD and its suppliers from software reuse should be considered. The following recommendations are made.

- 1) The ASSC should consider whether there is sufficient information to conclude that software reuse has potential benefits for the MOD and its suppliers and whether further effort is required to clarify the situation.
- 2) Many of the results of the US initiatives funded by the DoD are available in the public domain. The ASSC should consider whether it can assist in the dissemination of this information and whether it should encourage any form of UK participation.
- 3) If the MOD wishes to encourage software reuse, it should adopt an approach to procurement which is compatible with software reuse. The ASSC should consider whether it can assist the MOD to identify any aspect of the existing approach to procurement which is not supportive of increased software reuse, including standards relating to software.

## 7 REFERENCES

Shelburne B.J., Pitarys M.J.

Avionics software reusability observations and recommendations  
 Proceedings of the IEEE 1991 National Aerospace and Electronics Conference  
 NAECON 1991 IEEE Cat No 90CH3007-2, Dayton, OH, USA, Vol 2, May 1991,  
 pp614-19

Batory D., Coglianese L., Goodwin M., Shafer S.

Creating reference architectures: an example from avionics  
 SSR'95: Symposium on Software Reusability, Seattle, WA, USA, April 1995  
 SIGSOFT Software Engineering Notes, Spec issue, Aug 1995, pp27-37

Roskopf A.

Reusable input/output packages for Ada avionic applications  
 Ada: The choice for '92, Ada-Europe Interational Conference Proceedings,  
 Athens, Greece, May 1991, pp338-60

Gingerich T., Daniel W., Rasmussen K.

Reusable common module software for embedded systems  
 Proceedings of the IEEE 1989 National Aerospace and Electronics Conference  
 NAECON 1989, Dayton, OH, USA, 22-26 May 1989, Vol 4, pp1831-7

Coglianese L., Smith R., Tracz, W

DSSA case study: navigation, guidance and flight director design and  
 development  
 IEEE Symposium on Computer-Aided Control Systems Design (CACSD), Napa,  
 CA, USA, March 1992, pp102-9

Svboda F.

Reuse-based reengineering : notes from the underground  
 Proceedings of the Fourth Systems Reengineering Technology Workshop, APL  
 RMI-94-003, Monterey, CA, USA, Feb 1994, pp355-60

Smith M., Sodhi J.

Marching towards a software reuse future  
 Ada Letters, Vol 14, No 6, Nov-Dec 1994, pp62-72

Barlin B., Lawler J.M.

Effective software reuse in an embedded real-time system  
 TRI-Ada '92 Proceedings, Orlando, Fl, USA, Nov 1992, pp281-7

Juttelstad D., Stevens B., Rumbut J.

Ada reuse with the US Navy undersea application domain

Ada in Transition, 1992 Ada UK International Conference Proceedings, London, UK, Oct 1992, pp143-51

Ruegsegger T.B.

Making reuse pay: the SIDPERS-3 RAPID Center

IEEE Communications Magazine, Vol 26, No 8, Aug 1988, pp16-24

Ruegsegger T.

The RAPID center: a model for software reuse policy

Empirical Foundations of Information and Software Science V, 1990, pp455-63

Henry E., Faller B.

Large-scale industrial reuse to reduce cost and cycle time

IEEE Software, Vol 12, No 5, Sep 1995, pp47-53

Kirby J.M.

System development-the challenges of re-useability

Microprocessors and Microsystems, Vol 19, No 9, Nov 1995, pp503-10

McGarry F.

Experimental software engineering packaging for reuse

Experimental Software Engineering Issues: Critical Assessment and Future Directions, Dagstuhl Castle, Germany, Sep 1992, International Workshop Proceedings, pp213-15

Krutz W.K., Allen K., Olivier D.P.

The costs related to making software reusable: experience from a real project

Proceedings TRI-Ada '91, San Jose, CA, USA, 21-25 Oct 1991, pp437-43

Belanger D., Krishnamurthy B.

Practical software reuse: an interim report

Proceedings of 1994 3rd International Conference on Software Reuse, Rio de Janeiro, Brazil, Nov 1994, pp53-63

Joos R.

Software reuse at Motorola

IEEE Software, Vol 11, No 5, Sep 1994, pp42-47

Griss M.L.  
Software reuse experience at Hewlett-Packard  
Proceedings of 16th International Conference on Software Engineering, Sorrento,  
Italy, May 1994, p270

Fafchamps D.  
Organisational factors and reuse  
IEEE Software, Vol 11, No 5, Sep 1994, pp31-41

Lim W.C.  
Effect of reuse on quality, productivity and economics  
IEEE Software Vol 11, No 5, Sep 1994, pp23-30

Bauer D.  
A reusable parts center  
IBM Systems Journal, Vol 32, No 4, 1993, pp620-624

Tirso J.R., Gregorius H.  
Management of reuse at IBM  
IBM Systems Journal, Vol 32, No 4, 1993, pp612-15

Rosenbaum S., du Castel B.  
Managing software reuse-an experience report  
Proceedings ICSE-17 Workshop on Formal Methods Application in Software  
Engineering Practice, Seattle, WA, USA, Apr 1995, pp105-111

Hutchinson J.W., Hindley P.G.  
A preliminary study of large-scale software reuse  
Software Engineering Journal, Vol 3, No 5, Sep 1988, pp208-12

Anderson C.M.  
Reusable software-a mission critical case study  
Proceedings of COMPAC 85, The IEEE Computer Society's Ninth International  
Computer Software and Applications Conference IEEE Cat No 85CH2221-0,  
Chicago, IL, USA, Oct 1985, p205

Beser E., Williamson J.S.  
Reusable Ada avionics software packages library system  
Proceedings of the IEEE 1992 National Aerospace and Electronics Conference,  
NAECON 1992 IEEE Cat No 92CH3158-3, Dayton, OH, USA, Vol 2, May 1992,  
pp598-602

Mullins B.E.  
Common Ada missile packages (CAMP)  
Software for Guidance and Control (AGARD-CP-503), Thessaloniki, Greece,  
May 1991, pp24/1-6

Coomer T.N. Jr, Comer J.R., Rodjak D.J.  
Developing reusable software for military systems-why it is needed why it isn't  
working  
SIGSOFT Software Engineering Notes, Vol 15, no 3, July 1990, pp33-8

Frakes W.B., Biggerstaff T.J., Prieto-Diaz R., Matsumura K., Schaefer W.  
Software reuse: is it delivering?  
13th International Conference on Software Engineering, Austin, TX, USA, May  
1991, pp52-59

Moore J.W.  
The evolving role of software reuse  
TRI-Ada '92 Proceedings, Orlando, Fl, USA, Nov 1992, pp268-75

Neighbors J.M.  
Reuse so far: phasing in a revolution  
Proceedings of 1994 3rd International Conference on Software Reuse, Rio de  
Janeiro, Brazil, Nov 1994, pp191-2

Tracz W.  
Software reuse myths revisited  
Proceedings of 16th International Conference on Software Engineering, Sorrento,  
Italy, May 1994, pp271-2

Tracz W.  
Reuse state of the art and state of the practice report card  
Proceedings of 1994 3rd International Conference on Software Reuse, Rio do  
Janeiro, Brazil, Nov 1994, pp193-4

Basili V.R.

Facts and myths affecting software reuse

Proceedings of 16th International Conference on Software Engineering, Sorrento, Italy, May 1994, pp269

Poulin J.S., Caruso J.M., Hancock D.R.

The business case for software reuse

IBM Systems Journal, Vol 32, No 4, 1993, pp567-94

Morrison J.S.

Strategic defense initiative (SDI) software reuse issues

Technology of Object-Oriented Languages and Systems, Proceeding of the Second International Conference, TOOLS, Paris, France, 25-29 Jun1990, pp419-31

Moore J.W.

A structure for a defence software reuse marketplace

Ada Letters, Vol 14, no 3, May-Jun 1994, pp80-90

Cardow J.E.

Issues on software reuse

Proceedings of the IEEE 1989 National Aerospace and Electronics Conference NAECON 1989, Dayton, OH, USA, May 1989, Vol 2, pp564-567

Huber T.R.

Reducing business and legal risks in software reuse libraries

Proceedings of Third International Conference on Software Reuse: Advances in Software Reusability, Rio de Janeiro, Brazil, Nov 1994, pp110-17

Gowda, R.G., Satterthwaite C.P.

Management issues in developing reusable avionics software

Proceedings of the IEEE 1994 National Aerospace and Electronics Conference NAECON 1994, Dayton, OH, USA, May 1994, Vol 2, pp866-73

Tirso J.R.

Establishing a software reuse support structure

ICC 91 International Conference on Communications, Denver, Co, USA, Jun 91, pp1500-4 vol 3

Jahanian F.

Proceedings of the Fourth Systems Reengineering Technology Workshop, APL RMI-94-003, Monterey, CA, USA, Feb 1994, pp91-6

Benjamin C.L., Pitarys M.J., Solomon E.N., Sedrel S.

A common Ada run-time system for avionics software

AGARD Conference Proceedings 545, Aerospace Software Engineering for Advanced Systems Architectures, Paris, France, May 1993, p16/1-11

Poulin J.S.

Debate on software reuse libraries

Proceedings of 1994 3rd International Conference on Software Reuse, Rio de Janeiro, Brazil, Nov 1994, p202

Skoglund R., Johnson J.

Code reuse in Ada and C++

Embedded Systems Programming Vol 6, No 5, May 1993, pp44-8,50,52,54

Shaw M.

Architectural issues in software reuse: it's not just the functionality, it's the packaging

SSR'95: Symposium on Software Reusability, Seattle, WA, USA, Apr 1995, SIGSOFT Software Engineering Notes, spec.issue, Aug 95, pp3-6

Garland D., Allen R., Ockerbloom J.

Architectural mismatch: Why reuse is so hard

IEEE Software, Vol 12, No 6, Nov 1995, pp17-26

Zand M.K., Samadzadeh M.H.

Software reuse issues perspectives

IEEE Potentials, Vol 13, No 3, Aug-Sep 1994, pp15-19

Favaro J.

Future directions for reuse in Europe

Proceedings of 1994 3rd International Conference on Software Reuse, Rio de Janeiro, Brazil, Nov 1994, pp221-2

Frakes W.

The future of systematic software reuse

Proceedings of 1994 3rd International Conference on Software Reuse, Rio de Janeiro, Brazil, Nov 1994, p220

Meyer W., Lausecker H.

Re-usable software: challenge for the future

Conference on Applied Computer Science and Software, Turning Theory into Practice, Munich, Germany, Sep-Oct 1991, pp136-47

Potter M.R.

DoD software master plan: past studies on software issues

Conference proceedings, Third International Software for Strategic Systems, Huntsville, AL, USA, Feb 1990, p184

DoD Software Reuse Initiative. Vision and Strategy

9th Edition, February 12, 1996 Document No: PD134.8:12-FEB-96

URL <http://sw-eng.falls-church.va.us/reuseic/history/visstrat/2-12-96.html>

**Prepared by:**

**K John-Phillip, D W R Marsh**  
**Software and Systems Integrity Department**  
**ERA Technology**