NASA-CR-193114

The Repository-Based Software Engineering Program:

Redefining AdaNET as a Mainstream NASA Resource

GRANT
IN-61-CR
167277
p. 18

Applied Expertise, Inc.

2/9/93

(NASA-CR-193114) THE REPOSITORY-BASED SOFTWARE ENGINEERING PROGRAM: REDEFINING AdaNET AS A MAINSTREAM NASA SOURCE (Research Inst. for Computing and Information Systems) 18 p

N93-28328

Unclas

G3/61 0167277

Cooperative Agreement NCC 9-16
Research Activity No. RB.04

NASA Technology Utilization Program
NASA Headquarters

Research Institute for Computing and Information Systems
University of Houston-Clear Lake

WHITE PAPER

The RICIS Concept

The University of Houston-Clear Lake established the Research Institute for Computing and Information Systems (RICIS) in 1986 to encourage the NASA Johnson Space Center (JSC) and local industry to actively support research in the computing and information sciences. As part of this endeavor, UHCL proposed a partnership with JSC to jointly define and manage an integrated program of research in advanced data processing technology needed for JSC's main missions, including administrative, engineering and science responsibilities. JSC agreed and entered into a continuing cooperative agreement with UHCL beginning in May 1986, to jointly plan and execute such research through RICIS. Additionally, under Cooperative Agreement NCC 9-16, computing and educational facilities are shared by the two institutions to conduct the research.

The UHCL/RICIS mission is to conduct, coordinate, and disseminate research and professional level education in computing and information systems to serve the needs of the government, industry, community and academia. RICIS combines resources of UHCL and its gateway affiliates to research and develop materials, prototypes and publications on topics of mutual interest to its sponsors and researchers. Within UHCL, the mission is being implemented through interdisciplinary involvement of faculty and students from each of the four schools: Business and Public Administration, Education, Human Sciences and Humanities, and Natural and Applied Sciences. RICIS also collaborates with industry in a companion program. This program is focused on serving the research and advanced development needs of industry.

Moreover, UHCL established relationships with other universities and research organizations, having common research interests, to provide additional sources of expertise to conduct needed research. For example, UHCL has entered into a special partnership with Texas A&M University to help oversee RICIS research and education programs, while other research organizations are involved via the "gateway" concept.

A major role of RICIS then is to find the best match of sponsors, researchers and research objectives to advance knowledge in the computing and information sciences. RICIS, working jointly with its sponsors, advises on research needs, recommends principals for conducting the research, provides technical and administrative support to coordinate the research and integrates technical results into the goals of UHCL, NASA/JSC and industry.

RICIS Preface

This research was conducted under auspices of the Research Institute for Computing and Information Systems by Applied Expertise, Inc., Dave Dikel acting as Principal Investigator. Dr. E. T. Dickerson served as RICIS research coordinator.

Funding was provided by the NASA Technology Utilization Program, NASA Headquarters, Code C, through Cooperative Agreement NCC 9-16 between the NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA research coordinator for this activity was Ernest M. Fridge III, Deputy Chief of the Software Technology Branch, Information Technology Division, Information Systems Directorate, NASA/JSC.

University of Houston - Clear Lake
Repository-based Software Engineering Program

White Paper

THE REPOSITORY-BASED SOFTWARE ENGINEERING PROGRAM: REDEFINING ADANET AS A MAINSTREAM NASA RESOURCE

February 9, 1993

NASA Cooperative Agreement No. 9-16, Subcontract No. 101
Sponsored by NASA Technology Transfer Division

White Paper

The Repository-based Software Engineering Program: Redefining AdaNET as a Mainstream NASA Resource

At the National Aeronautics and Space Administration (NASA), software has been a key element in nearly every project. In recent years, both the criticality and complexity of NASA software have been increasing at rates that demand new, more effective software development practices; otherwise, the software element of a project may cause its failure.

- "Software Engineering Program Plan," NASA Office of Safety and Mission Quality, April 1992.

Scarce resources, fiercely competing national priorities and NASA's growing need for reliable software call for innovative approaches to the way NASA develops software. In software development, as in other engineering disciplines, common approaches to similar types of problems help to ensure reliability and avoid unnecessary expense. Commonality also reduces the complexity of integrating, testing and upgrading software systems that must work together. Even though many NASA programs are now developed by multiple contractors and managed by more than one center, there is little commonality or standardization in software development practices across NASA. Further, NASA-developed software technologies that could provide valuable solutions throughout the agency remain isolated. As a result, unnecessary costs are incurred in a time of scarce resources and competing demands, and quality is less assured.

Solving these problems is not simple. It requires technological innovation and cultural change. NASA's Repository-based Software Engineering Program, together with other programs, is taking focused steps toward promoting NASA's appropriate use of common standards and guidelines, and reuse of software models, practices and components. An effective reuse program promotes common standards and practices, improves software quality and reduces life-cycle cost.

Purpose and Scope

This white paper is intended to inform and update senior NASA managers about the Repository-based Software Engineering Program (RBSE). The paper provides background and historical perspective on software reuse and RBSE for NASA managers who may not be familiar with these topics. The paper draws upon and updates information from the RBSE Concept Document, baselined by NASA Headquarters, Johnson Space Center and the University of Houston - Clear Lake in April 1992.

The substance of the paper describes several of NASA's software problems and what RBSE is now doing to address those problems. The paper also provides next steps to be taken to derive greater benefit from this Congressionally-mandated program. The section on next steps describes the need to work closely with other NASA software quality, technology transfer, and reuse activities and focuses on goals and objectives relative to this need.

This paper focuses on RBSE's role within NASA; however, there is also the potential for systematic transfer of technology outside of NASA in later stages of the RBSE program. This technology transfer is discussed briefly, and by no means comprehensively, in a note at the end of this paper.

Background

The RBSE Program was initiated in 1991 as the result of a redirection and restructuring of the AdaNET repository operated in West Virginia. RBSE is dedicated to introducing effective software reuse into the mainstream practice of its customer base. This section introduces the concept of reuse and familiarizes the reader with RBSE.

Software Reuse and Why It's Important

The Department of Defense defines reuse as, "The application of a reusable [software] component to more than one application system." Reuse can occur within a system, such as within a robot arm; across similar systems, like a product line of household appliances; or in common services for widely different systems, for example, user interface tools. Any product created during the software development life-cycle[1] is considered a component, and a potential resource for reuse.[2] We emphasize the broadest interpretation of this definition so that a component can also refer to any artifact (e.g. process definition) that enables a work product or process to be reused during the development of a system.

Most engineering disciplines re-use common practices, architectures, processes, standards and off-the-shelf parts to build products. This is much less the case with software. Grady Booch, a top industry expert who specializes in the design of very large software systems, points out that few construction companies build on-site steel mills to forge custom girders, yet in the software industry, such practice is common.[3]

Current software development practices lack the clarity, consistency and predictability that other, more mature engineering disciplines provide as a matter of course. Nevertheless, software is critical to the missions of virtually all private and public organizations. A recent Business Week article states "High quality software is key to running everything from personal computers to Patriot missiles."[4]

In answer to this need, there is growing evidence that *when reuse efforts are "carefully planned and properly carried out,"*[5] costs are reduced and quality is increased. Following are several real-life examples that illustrate this point:

- At a recent conference, Dr. Donald Mullikin, Deputy Program Manager of the U.S. Federal Aviation Administration's Advanced Automation System (AAS) described significant overall gains in productivity associated with reuse of about 1.3 millions lines of code from one operational AAS system to another. He attributed this to the program's organized reuse effort.[6]
- NASA's own Software Engineering Laboratory (SEL) has demonstrated gains in productivity associated with reuse. Located at NASA's Goddard Space Flight Center, the SEL has kept extensive data on over 90 projects for about 16 years. These projects, typically involving 15-25 people, developed operational systems for NASA's Flight Dynamics Division. Recent SEL studies show that benefits realized by increasing software reuse include a reduction in cost to develop the software by one-third and an improvement in reliability (as measured by defect rates) of 87 per cent.[7]
- A Hewlett-Packard study documented that projects within one division that did not reuse software components took 60 percent longer than those that did. In addition, when reuse was built into the design of an entire product line, pre-release defects dropped from more than 180 in the first product to less than ten in the tenth and subsequent products.[8]

Getting Effective Reuse Into Practice is Serious Business

Getting software development and reuse to the level of other engineering practices requires a long-term commitment and effective action by the organization that seeks its benefits. For example, Dr. John Foreman, head of DoD's Software Technology for Adaptable, Reliable Systems (STARS) Program, stated at a recent conference that implementing reuse is not simple; it involves changing the way an organization does business. The STARS Program, for example, is addressing management and technology transition as well as technological issues to "enable a paradigm shift" for its customers.[9] Similarly, DoD's Vision and Strategy seeks to "ensure that reuse is treated as an inseparable part of software engineering," and lays out key goals, vision and strategy to produce long-term reuse benefits for DoD[10].

Reuse requires management support, just as does any other substantial quality improvement initiative. Software must be designed and built with reuse in mind to maximize benefits, even though this may add initial costs. For example, Goddard's Software Engineering Laboratory involves a separate organization to capture, analyze, package and tailor experience as well as software for reuse.[11] While organizations may see real improvements right away, Frank McGarry, Head of Goddard's Software Engineering Branch, and co-founder of the SEL, stresses that significant improvements or cost savings can take time. For example, the first measurable impact of a comprehensive reuse/process improvement effort, such as the SEL's Experience Factory, may be several years away from its start.[12]

RBSE and Its Accomplishments

The RBSE program supports an operational software repository (reuse library) and a small, expert research staff. The repository acquires and distributes "assets" -- software components for reuse and related software engineering information to support their effective reuse.

In September 1992, RBSE fielded an improved operational reuse library facility. The new system was adapted from Johnson Space Center's NASA Electronic Library System (NELS) and represents a significant advance in the program's technical capabilities. The library now can be accessed from several platforms including X Windows and DOS. It provides the user with multiple search methods and supports a wide range of software engineering assets. Expanded tracking capabilities provide a detailed accounting of product distribution. These technical improvements make RBSE's facilities competitive with any other major government-funded repository. A number of reuse efforts external to NASA have expressed interest in acquiring RBSE's system software.

RBSE has two primary activities: library operations and research. The library operations activity provides a broad range of "public domain" reusable software, including subsystems and tools, and technical information on publications, contracts, conferences, products and training resources. An experienced and competent staff provides information services, including a help desk.

During Calendar Year 1992, RBSE library operations delivered over 2,900 "objects" which ranged from software components to journal abstracts to a wide variety of customers nationwide. As of January 25, 1993, there were 623 customers.[13] RBSE has enabled the introduction of reuse into the curriculum at West Virginia University and the University of Houston by providing a repository of software components at little or no cost.

RBSE's research activity consists of a small group, skilled in the areas of software engineering process and data modeling; domain engineering; generic software architectures; data-base design; knowledge-based systems; Object-oriented methods; and mission critical software. RBSE applies its research capabilities to help its customers apply reuse technology to solve real-world problems. For example, in the Space Shuttle pilot (discussed below, see STSOC) RBSE's research activity is assisting programmers in transforming existing FORTRAN code using Object-oriented methods.

The program has also taken a leadership role in several nationally-recognized efforts to improve software quality through reuse. For example, RBSE hosted a workshop on the Software Engineering Institute's Design for Reuse Handbook and co-hosted the Reuse Education Workshop with DARPA and Air Force Reuse efforts. (See Liaison Activities section for additional information.)

New Focus on NASA

In late 1992, the NASA Level 1 Program Manager directed RBSE to sharpen focus and maximize the impact of scarce resources by concentrating its efforts on internal NASA customers. In our view, there is a clear need for an institution devoted to promoting and disseminating reusable software technology within NASA. RBSE is now uniquely positioned to play a substantial role in addressing this need. RBSE's role, current program activities and next steps are discussed below.

NASA Software Development Problems

Software is critical to the success, and often the safety, of nearly all NASA programs. A recent study by NASA of its software estimated that, "development, purchase and maintenance of software require more than 20 per cent of the total NASA budget." The same study stated that, "software is a critical component of nearly all NASA projects," and that, "software's size, influence and criticality are growing dramatically within NASA."[14]

As illustrated below, there are a number of compelling software development problems that fall within the scope of the RBSE program.

Common Standards and Approaches Are Lacking

In software development, as in other engineering disciplines, common approaches (e.g. practices, architectures, off-the-shelf parts) to similar types of problems help to ensure software reliability and avoid unnecessary expense. Standards reduce the complexity of integrating, testing, upgrading and maintaining software systems. In a similar way, the use of common software architectures encourages systematic software reuse. Commonality also reduces the complexity of integrating, testing and upgrading software systems that must work together. Without common standards and practices, costs are driven up because NASA and its contractors must maintain (and sometimes reinvent) multiple sets of practices, processes and development environments and train staff to work effectively within each set. This also inhibits the sharing of people across different projects.

In addition, a commonly used set of standards and guidance for their use can provide the basis for the appropriate application and tailoring of pre-existing solutions to a specific problem, project or organization.[15] Yet there is little commonality or standardization in software development practices across NASA.[16] The 1989 NASA-wide software study found that, "NASA does not have an adequate set of agency-level standards and internal organizations to provide direction in software engineering...."[17]

The study notes that, even though projects often span multiple centers and NASA contractors often support multiple projects, NASA centers and individual projects are left to establish their own software standards and practices. As a result, "many different groups within the agency must deal with the same issues," and contractors who deal with several parts of the agency often must work with multiple sets of standards.[18] Several contractors also commented that NASA's lack of software standards dramatically increased the costs of doing business with NASA.[19]

The lack of common practices also results in an inefficient use of resources and project time.[20] For example, a recent report by the U.S. General Accounting Office found that "developers [from multiple companies and locations] are writing [Space Station] flight software without needed software standards....As a result, NASA's strategy for controlling costs has been badly weakened." [21] After reviewing the report, the Chairman of the U.S. House of Representatives Committee on Science, Space and Technology - Subcommittee on Investigations and Oversight expressed concern over NASA's lack of software standards. The chairman stated that "we have contractors performing software development while lacking guidance on the program goals, objectives, technical and management approaches, performance expectations ... in short, almost everything NASA needs to assure that software has been properly developed."[22]

Software Technology Innovations Remain in "Pockets"

In addition to NASA's lack of common software standards and approaches, many potentially valuable solutions are isolated by a lack of communication. Software innovations do not move easily, if at all, from one NASA organization to another. A number of sources point to a lack of communication or exchange between different development teams, resulting in isolated pockets of software technology. For example:

- Even when labs or projects develop or adapt technology that promises to save effort and expense agency-wide, they often have difficulty disseminating the technology within NASA.[23]
- While a process exists to distribute tools and finished software within NASA, no formal mechanisms exist to facilitate the transfer of software engineering technology (e.g. processes, methods and techniques).[24]
- Labs often have difficulty discovering which software research issues are critical to projects.[25]

Formal Mechanisms that Promote Reuse of NASA-Use-Only Software Are Limited

The primary formal mechanism that now exists to address NASA software reuse is NASA Management Instruction (NMI) 2210.2B, which designates COSMIC as the distributor of software that is cleared for public release, documented and ready to run. COSMIC distributes software to NASA and its contractors at no charge in exchange for rights to distribute the software to others. However, current provisions for disseminating NASA-developed software fail to adequately address transfer of non-public domain software from NASA developer to NASA developer. COSMIC is not funded to distribute software that the government has rights to use but not distribute; software that cannot be distributed for proprietary or security reasons; or software life-cycle components that may be precursors to or parts of finished products. While NASA policy does not preclude NASA developers from sharing government-owned software with one-another, no organizational structure or operational unit is in place to promote this transfer.

A possible pathway for distributing NASA-use-only software is the High Performance Computing and Communications (HPCC) Initiative's Software Exchange. NASA has a leading role in creating this Software Exchange among the software libraries of sponsoring agencies. The Software Exchange could be leveraged to facilitate the distribution of NASA-use-only software; however, a number of proprietary, security and technical issues need to be addressed by participating libraries.

RBSE'S CURRENT ACTIVITIES

The problems described above cannot be fully addressed by any one program. RBSE resources are limited, as is its current sphere of influence. Similarly, software reuse technology, while it is seen by major organizations such as DoD as a key to solving the types of problems described above, is not fully mature. Finally, solving these problems will involve major cultural changes and require significant commitments at all levels of NASA. Nevertheless, RBSE is currently taking focused steps to effect change in concert with several key NASA activities, including:

- Demonstrating reuse viability within a major Space Transportation System (Shuttle) Operations Contract pilot program
- Supporting NASA-wide standardization efforts by providing library resources to the Strategic Avionics Technologies Working Group
- Promoting common approaches and sharing information by participating in the Software Technology Working Group
- Maintaining liaison with key elements of the software engineering community

By discovering "real world" customer requirements, and establishing the basis for long-term customer relationships, these activities may provide an initial approach to solutions for the major NASA-wide problems stated above.

Space Transportation System (Shuttle) Operations Contract Pilot Program

The Space Transportation System Operations Contract (STSOC) Pilot Program is a series of pilot projects that seeks to upgrade about 10 million lines of software code. The "preliminary" pilot is a proof of concept for techniques to convert unstructured FORTRAN flight design code to Object-oriented software. The premise is that the rewrite will reduce the total number of lines of code and will cost significantly less to maintain. RBSE will provide training in Process Modeling, Object-oriented Analysis and Domain Engineering to transition FORTRAN programmers to Object-oriented Design and Data Base technology. RBSE will define enactable process descriptions; collect and catalog reusable software and related information; and promote the use of these assets within the pilot. RBSE will also train and support pilot staff to make effective use of RBSE's reuse library.

If the preliminary pilot succeeds, the next step is an expanded pilot that will involve converting about 2 million lines of code. The success of the expanded pilot could then lead to the upgrade of the full 10 million lines of code.

By providing consulting services to the preliminary pilot effort, RBSE will help adapt software reuse technologies to the needs of NASA, demonstrate their viability and develop a strategic capability. RBSE's collection of artifacts, data and lessons learned in STSOC's transition to Reusable Object-oriented Software Engineering (ROSE) provides specific assistance to STSOC management, as well as assets for the NASA community at large.

Strategic Avionics Technology Working Group

The Strategic Avionics Technology Working Group (SATWG) is a NASA-wide collaboration formed to coordinate Research and Development in avionics and provide a dialog between users and suppliers of NASA space avionics technology. Its challenge is to "deploy advanced methods of systems engineering that will span multiple NASA programs and lead to smarter, faster and more cost effective programs." The SATWG involves all nine NASA field centers, the major U.S. integrating prime contractors, most second tier avionics contractors, and representatives from the DoD and academia.[26] In a recent memo, a senior SATWG official listed software reuse and the use of Object-oriented design as priorities in the requirements for avionics technology.[27]

The SATWG has invited RBSE to explore a potential support role. This provides an opportunity to adapt RBSE operations to serve a group working to improve NASA software. RBSE can leverage its capabilities to promote NASA-wide standards and common approaches to software development. At the same time, RBSE can build a solid customer base from the NASA field centers and major prime contractors.

Software Technology Working Group

A number of NASA centers such as Langley Research Center (LaRC), Johnson Space Center and the Jet Propulsion Laboratory are driving a NASA-wide effort to coordinate software technology research activities. This effort responds to NASA labs' uncertainty about Headquarters' support for funding and coordination of software engineering research. The Software Technology Working Group (STWG) was organized to coordinate, focus and improve NASA software technology research efforts to increase NASA's ability to develop "cost-effective, high quality software through reuse."[28] The group's focus has recently narrowed to, "increase the technology transfer impact of NASA research activities."[29]

RBSE is a founding member of the STWG. It has coordinated local arrangements and will host the July STWG meeting jointly with Johnson Space Center. In this way, RBSE is helping to foster communications between and among research facilities and their customers in mission programs.

RBSE Liaison Activities

No one program can address all the interrelated and entrenched technical and non-technical barriers that stand in the way of implementing effective software reuse practices. This requires coordination, cooperation and teamwork. RBSE is working to leverage the efforts of related reuse programs within organizations such as DoD, as well as maintaining liaison with related NASA activities.

RBSE and other repositories can greatly increase their "added value" by cooperating amongst themselves and by supporting efforts that help repository customers to change the way they develop software. While a repository operating passively, by itself, encourages "opportunistic" (ad hoc) reuse, it *cannot* fundamentally change the way its customers develop software -- often, in effect, the way they do business. Such changes are often required to achieve the gains in quality and productivity that is attributed to software reuse.

By banding together, repositories can increase their supplier and customer base and benefit from research activities that many reuse libraries currently fund. For example, the Air Force CARDS Program is developing an Acquisition Handbook to guide managers who want to encourage reuse through the Federal DoD Acquisition Regulation (DFAR).

RBSE actively supports the Reuse Library Interoperability Group (RIG). The RIG is comprised of over 30 organizations, each with a stake in the success of a reuse library effort. Members include government agencies, reuse libraries and commercial firms. The RIG's primary purpose is to facilitate the efficient, effective sharing of assets and other information among government software reuse libraries.[30] RBSE also works with the Reuse Acquisition Action Team, a group that supports DoD's software reuse Management Issues Working Group in their efforts to identify and remove non-technical barriers to reuse.

These liaison activities are important because they can increase RBSE's customer and supplier base, provide critical management and technical information from the reuse initiatives of other Federal and commercial organizations, and help to identify areas of overlap and potential collaboration with other reuse programs.

Success Criteria for RBSE Current Activities

RBSE's near-term objective is to demonstrate unique value-added capabilities to both STSOC and SATWG. Achievement of this objective will be evidenced by meeting one or more of the following criteria:

- Endorsement by the STSOC Program Manager
- Support of RBSE's inclusion in the follow-on STSOC pilot
- Regular use of RBSE by major SATWG contractors and NASA centers
- Formalized role for RBSE in support of SATWG

The immediate goal of RBSE's STWG activities is to jointly achieve Headquarters' support for the STWG as a forum for coordinating software research and advancing software technology practice within NASA.

NEXT STEPS

To establish a clear, visible role within NASA's research and technology transfer activities, the RBSE Program is considering the following next steps:

- Working closely with other NASA software quality, technology transfer and reuse activities to ensure that RBSE addresses NASA problems that matter
- Providing a gateway and mechanism for NASA to obtain software reuse technology and promote innovative ways to remove barriers to its effective implementation

Working Closely with other NASA Software Quality, Technology Transfer and Reuse Activities

Working closely with other NASA reuse-problem-solving activities will ensure that RBSE has a clear and current knowledge of NASA's software engineering/reuse problems and needs. This will sharpen RBSE's ability to provide NASA with products and services that solve vital problems. Further, RBSE can leverage its effectiveness and impact by providing a specific set of services directly to these activities. The following three activities are prime targets for collaboration.

1. The Code QE (Quality Engineering) Software Engineering Program appears to offer the most leverage. The Program is systematically addressing NASA's lack of common policies, standards and practices and NASA's need to capture and share improvements and innovations among its software developers.[31] RBSE can increase its impact by working closely with this program.

   For example, the QE Program's "Experience Factory" Task offers a number of opportunities RBSE collaboration and service. The Task seeks to develop the Experience Factory concept and establish the process for implementing the concept across NASA. The Experience Factory concept was developed at Goddard's Software Engineering Lab (SEL) by Dr. Victor Basili to "institutionalize the collective learning of an organization that is at the root of continual improvement and competitive advantage."[32] Among other things, it stresses the identification, packaging and reuse of software-related products and experience. *One of the principle goals of an experience factory is "to make reuse easy and effective."*[33] The Experience Factory concept has evolved over 16 years and demonstrated measured results for NASA Goddard Space Flight Center, Flight Dynamics Division.

   RBSE could work with Code Q to evolve the experience factory concept and help others to apply it NASA-wide. RBSE's role within the STSOC pilot will produce a hands-on understanding of the concept, since RBSE is involved with identifying, packaging, storing and facilitating the reuse of software, metrics and other valuable experience. Also, RBSE can provide a hub for Experience Factory information. RBSE can collect packaged experience (e.g. lessons learned, design-development-test process, software components) from Experience Factory efforts and disseminate them within NASA. In addition, the Experience Factory Task activity includes a survey to establish the baseline characteristics of NASA's software and software engineering process. Study findings may help NASA's suppliers and developers to identify reuse barriers and opportunities within NASA.

2. The Software Technology Working Group (described above) promises to take a leading role in coordinating the software research among NASA centers to increase the transfer and impact of software technology. Along with coordinating research, the STWG also plans to identify the needs of internal and external customers and facilitate the insertion of software technology. The group is likely to require a vehicle to promote products and acquire feedback throughout NASA, and RBSE may fill this role. While working to evaluate the STWG as a potential customer, RBSE will also come into contact with many potential customers for its other products and services.

3. The HPCC Software Exchange (described above) could be used by RBSE to expand the distribution of its products and services both within and outside NASA. While initial discussions have taken place, the details need to be defined and implemented.

Providing a Gateway for NASA

Providing a gateway between NASA and other reuse initiatives such as those within the Department of Defense, can alert NASA managers to practical solutions and hidden pitfalls. RBSE can acquire and disseminate both software reuse technology and innovative strategies to remove barriers to its effective implementation. RBSE currently works to identify and acquire technology, tools and policy solutions rather than to reinvent them. For example, RBSE is working to remove key acquisition and management barriers to reuse in collaboration with the Department of Defense [Reuse] Management Issues Working Group. Similarly, RBSE is working with the Software Engineering Institute to develop effective approaches to designing for software reuse. RBSE is positioned to disseminate such information and insights throughout the NASA software engineering community.

CONCLUSION

Software reuse promises to be a key factor in addressing one of NASA's most critical risk areas which effects the cost, schedule and safety of virtually all NASA programs. NASA's current lack of common software standards and practices inhibits effective software reuse and increases cost and risk. This occurs during a time of shrinking budgets and increasing software requirements. In our view, the growing need for and critical nature of NASA's software requirements challenge NASA to adopt innovative approaches to software development. NASA can meet this challenge by promoting software reuse and by sanctioning NASA-wide software standards, common software development practices and formal software technology transfer mechanisms.

We believe RBSE can be a valuable part of the solution to NASA's software problems, and a potential resource for NASA software development activities. RBSE is already supporting NASA-wide standardization efforts by providing library resources to the Strategic Avionics Technology Working Group, and is currently implementing a pilot program demonstrating reuse viability within a major space transportation system (Shuttle).

RBSE is dedicated to introducing software reuse into its customers' mainstream software development practices. We believe it provides NASA with an established vehicle to achieve its self-imposed goal of developing safe, high-quality, cost-effective software.

Note on Technology Transfer to Industry

Many companies that produce large, complex software systems can benefit from software reuse. However, for the RBSE program to effectively transfer this technology, it must tailor its products and services to the needs of specific customers groups. By focusing on initial customers such as STSOC and SATWG, RBSE can separate products and services that have value from those that are "nice to have." The next step is to refine and develop a select set of products and to understand who uses these products, how they are used and what makes them valuable. During this process, RBSE can identify other customer groups with needs similar to its initial customers'. Once these groups are identified, RBSE can test and refine the commercial viability of its products and services.

For the next one to two years, RBSE will be engaged in primary targeted technology transfer, defined by a recent NASA study as , "the transfer of technology from a primary NASA mission ..., where the technology was developed and targeted for a particular customer segment ... from its inception."[34] For example, as part of the STSOC pilot RBSE is teaching Rockwell engineers Object-oriented Design specifically for their work on that project. RBSE is also learning what products and services it needs to provide to enable the transition to this new technology, as part of the process described above.

Over the following two or three years, RBSE could test and refine its products and services with customers whose needs are similar to the Rockwell engineers. The outcomes of this process should include transfer of reuse technology to other users within NASA and outside NASA, and the eventual commercialization of key products and services.

LIST OF ACRONYMS

AAS	:	U.S. Federal Aviation Administration's Advanced Automation System
CARDS	:	Central Archive for Reusable Defense Software
COSMIC	:	Computer Software Management and Information Center
DARPA	:	Defense Advanced Research Projects Agency
DFAR	:	Federal DoD Acquisition Regulation
DoD	:	Department of Defense
FAA	:	Federal Aviation Administration
HPCC	:	High Performance Computing and Communications Initiative
LaRC	:	Langley Research Center
NASA	:	National Aeronautics and Space Administration
NELS	:	Johnson Space Center's NASA Electronic Library System
NMI	:	NASA Management Instruction
QE	:	Quality Engineering
RBSE	:	Repository-based Software Engineering Program
RIG	:	Reuse Library Interoperability Group
ROSE	:	Reusable Object-oriented Software Engineering
SATWG	:	Strategic Avionics Technology Working Group
SEL	:	Software Engineering Laboratory at NASA's Goddard Space Flight Center
STARS	:	DoD's Software Technology for Adaptable, Reliable Systems Program
STSOC	:	Space Transportation System Operations Contract
STWG	:	Software Technology Working Group

NOTES

---

1 The software engineering life-cycle describes a series of stages for software development from the time a concept is defined through software implementation. NASA and the Defense Department generally adhere to the following eight-step life cycle:

1 - Concept and Initiation
2 - Requirements
3 - Architectural Design
4 - Detailed Design
5 - Implementation / Coordination
6 - Integration and Test
7 - Acceptance and Delivery
8 - Sustaining Engineering and Operations

2 "DoD Software Reuse Initiative - Vision and Strategy", July 15, 1992.

3 Booch, Grady, *Object Oriented Design with Applications*, Benjamin/Cummings, 1991.

4 "Can the U.S. Stay Ahead in Software?," Business Week, March 11, 1991.

5 Margono, Johan and Rhoads, Thomas E., "Software Reuse Economics: Cost-Benefit Analysis on a Large-Scale Ada Project," Proceedings: *14th International Conference on Software Engineering*, Melbourne, Australia, May 11-15, 1992. Note: Margono points out that "a software reuse effort, if not carefully planned and properly carried out, oftentimes becomes an inhibitor rather than a catalyst to software productivity and quality."

6 Mullikin, Donald, "Advanced Automation System (AAS) and Ada - Lessons Learned from a Program Management Perspective," Presentation to the Ninth Annual Washington Ada Symposium, July 1992.

7 McGarry, Frank and Waligora, Sharon, "Experiments in Software Engineering Technology: Recent Studies in the SEL (1990-1991), *Proceedings of the Sixteenth Annual NASA/Goddard Software Engineering Workshop, December 1991*.

8 Jost, J., "The Successful Introduction of Software Reuse Across an Entity," *Proceedings of the Sixteenth Annual NASA Goddard Software Engineering Workshop*, December 1991.

9 Foreman, John T. Presentation to STARS ' 92 conference, December 1992.

10 "DoD Software Reuse Initiative - Vision and Strategy", July 15, 1992.

11 Basili, Victor, "Experience Factory Fundamentals" *Seventeenth Annual Software Engineering Workshop, Special Tutorial Section, Starting an Experience Factory*, December 1992. The separate organization is referred to as an experience factory. Even though the experience factory is conceptually separate, individuals may simultaneously fill roles within the project and experience factory.

12 McGarry, Frank, "Establishing an Experience Factory - Laying the Foundation," *Seventeenth Annual Software Engineering Workshop, Special Tutorial Section, Starting an Experience Factory*, December 1992.

13 Data provided by MountainNet, Inc., Morgantown, WV, January 1993.

14 "Ada and Software Management in NASA: Assessment and Recommendations," -- A Report to the Information Resources Management Council by the Ada and Software Management Assessment Working Group, NASA, March 1989.

15 Basili, Victor R, "The Experience Factory: A Capability Based Software Organization, " Tutorial for University of Maryland Instructional Television System, October 1992. (Excerpted from the definition of engineering as it applies to software.)

16 McKay, Charles W., and Shankar, K.S., "Why NASA, Code R, Should Sponsor Advanced Research in Software Engineering: A White Paper," October 1992.

17 "Ada and Software Management in NASA: Assessment and Recommendations," NASA, March 1989.

18 Ibid.

19 Comments were made during a NASA-wide review of the draft of "Ada and Software Management in NASA: Assessment and Recommendations"

20 Ibid.

21 GAO, Space Station NASA's Software Development Approach Increases Safety and Cost Risks, June 1992.

22 Wolpe, Howard, Chairman, US House of Representatives Committee on Space, Science and Technology - Subcommittee on Investigations and Oversight, Letter to Daniel S. Goldin, August 5, 1992.

23 Preliminary finding of Code QE Software Tech Transfer Study (Described later in the paper) conveyed by a principal researcher.

24 Ibid. According to the researcher, NASA's COSMIC repository's function is to support NASA-developed finished software *products*. It is not funded to provide services such as training and support that are necessary to transfer software *practices*.

25 Author's notes from Software Reuse Tools Workshop, Research Triangle, North Carolina, May 1992 and Software Technology Working Group meeting, Hampton, Virginia, January 1993..

26 Cox, K. J. and Brown, D.C., "SATWG Considerations for Space Avionics," ERA Technology 1992 Avionics Conference And Exhibition, December 1992.

27 Internal NASA correspondence.

28 Taken from a view graph presented by LaRC's Wayne Bryant to the NASA Software Reuse Tools Workshop, Research Triangle, North Carolina, May 1992.

29 Author's notes from Software Technology Working Group meeting, Hampton, Virginia, January 1993.

30 Charter of the Reuse Library Interoperability Group, enacted June 14, 1991.

31 "Software Engineering Program Implementation Plan," NASA Office of Safety and Mission Quality, NASA Headquarters, April 1992. Note: The Implementation Plan details five objectives, three of which are referenced above, namely: (i)"...Establish a Process Improvement Program, based on engineering methods and the concept of the 'Experience Factory.' Package the technology baseline for various classes of software within NASA." (ii) "...Establish Policy and Standards for software development, management and assurance," and (iii) "Promote and accelerate technology transfer." The Experience Factory objective focuses on continuous quality improvement within an organization. In addition, activities are underway to identify domains within NASA within which Experience Factory products can be effectively shared.

32 Basili, Victor; Caldiera, Gianluigi; McGarry, Frank; Pajerski, Rose; Page, Gerald; and Waligora, Sharon, "The Software Engineering Laboratory - an Operational Software Experience Factory," Proceedings of the 14th International Conference on Software Engineering."

33 Basili, Victor R and Gianluigi, Caldiera, "A Reference Architecture for the Component Factory," ACM Transactions on Software Engineering and Methodology, Vol. I, No. 1, January 1992

34 "NASA Technology Transfer," Report of the Technology Transfer Team, December 1992.