



# Software Engineering Research/Developer Collaborations in 2005 (CI05)

Final Report

March 7, 2006

Tom Pressburger, ARC

## Table of Contents

1	Overview .....	2
2	Summary of technology provider/software development project collaborations .....	4
2.1	ARC: “Application of Software Cost Reduction (SCR) Tools and Methods to On-Orbit Crew Displays” .....	4
2.2	IVVF: “Infuse CodeSurfer into NASA Code S IV&V Process” .....	5
2.3	JPL: “Application of SpecTRM at JPL’s Advanced Project Design Team (TeamX)” .....	6
3	Paths to further infusion of the technologies .....	7
3.1	SCR .....	7
3.2	CodeSurfer, CodeSonar .....	7
3.3	SpecTRM .....	8
4	Technology transfer lessons learned .....	8
5	Acknowledgements .....	9
6	Acronyms .....	10
7	References .....	10
7.1	Collaboration Reports .....	10
7.2	Technologies .....	11
7.2.1	SCR .....	11
7.2.2	CodeSurfer and CodeSonar .....	11
7.2.3	SpecTRM .....	11

## 1 Overview

In CY 2005, three collaborations between software engineering technology providers and NASA software development personnel deployed three software engineering technologies on NASA development projects (a different technology on each project). For technology references, see Section 7.2. The main purposes were to benefit the projects, infuse the technologies if beneficial into NASA, and give feedback to the technology providers to improve the technologies. Each collaboration project produced a final report; for references, see Section 7.1. Section 2 of this report summarizes each project, drawing from the final reports and communications with the software developers and technology providers. Section 3 indicates paths to further infusion of the technologies into NASA practice. Section 4 summarizes some technology transfer lessons learned. Section 6 lists the acronyms used in this report.

Below we restate our success criteria from our proposal to NASA's Office of Safety and Mission Assurance – Software Assurance Research Program (OSMA SARP) to oversee the collaborations:

"We would like one of the main success criteria to be that the research products used in the collaborations are adopted for future software development by the teams (or organization). However, this is unrealistic for mid TRL-level research products that may lack productization, and it may be unrealistic for high TRL or even for commercial products (for example, the license fee may be too high for a single team to bear). Thus we have identified several other success criteria.

1. The success criteria of the collaboration projects funded under this proposal are met. This includes a positive rating for each product on the evaluation criteria metric.
2. The research product is adopted by the collaborating software development team for current use.
3. The research product is included in a list of recommended development practices at a NASA Center or by contractor.
4. The software development team using the product provides feedback, including performance data, to the research team to guide future development of the product.
5. Six months after the funded collaboration period, the research product is still being used by the development project or by a successor development project.
6. The researchers and consumers recommend to the CTO methods of making future versions of the research products available within NASA (for example, by Open Sourcing or by licensing the technology commercially or to organizations such as the Southwest Research Institute).
7. Independent of the success of the collaborations, "lessons learned" regarding the challenges and success factors for software development technology infusion within NASA."

A modification of 3 is "The research product is recommended for a branch, division, or directorate at a center". That is the statement for which column 3 applies in the table below.

Also relevant to judging the impact of the collaborations is the penetration factor (PF) used for SARP quarterly reviews:

- PF 8: Data passed back to project;
- PF 9: Results actually used by the project.

The following table summarizes the impacts of the technology in each collaboration regarding the penetration factor and success criteria as well as brief notes.

**Y** = Yes **A** = Anticipated in 2006 – 2007 timeframe

<b>Project</b>	<b>PF</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>Outcomes</b>
<b>ARC</b> SCR on ISS payload software	<b>8</b>	<b>Y</b>			<b>Y</b>			<b>Y</b>	Good exposure to the technology.
<b>IVVF</b> CodeSurfer on Flight Software	<b>9</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>A</b>		<b>Y</b>	CodeSonar found non-trivial defects, and its use recommended.
<b>JPL</b> SpecTRM to capture mission design rationale	<b>8</b>	<b>Y</b>			<b>Y</b>			<b>Y</b>	JPL hired the technology assistant, an MIT student. Journal paper in progress.

## 2 Summary of technology provider/software development project collaborations

This section briefly describes each collaboration: its objectives, what happened, its impact on the project, and the success criteria that were met.

### 2.1 *ARC: “Application of Software Cost Reduction (SCR) Tools and Methods to On-Orbit Crew Displays”*

The SCR technology, developed at the Naval Research Laboratory (NRL), provides tools and a method for developing, simulating, and analyzing formal requirements specifications. An SCR specification is represented in a tabular format and is based on a state-machine model. In addition to tools for consistency checking to detect syntax and type errors, missing cases, unwanted non-determinism, and tools to check application properties, such as safety properties, SCR also supports rapid construction of graphical user interfaces (GUIs) that simulate the target system’s interface, allowing for simulation of the required system behavior based on the underlying SCR specification.

The goal of this project was to apply SCR tools and methods to develop and validate a requirements specification of the display interface to an incubator. The incubator was to be a Space Station Biological Research Project (SSBRP) science payload.

The SCR technology providers gave a three-day training course on the SCR tools and method at the NASA Ames Research Center (ARC) to the project members. Lack of availability of the tool on the preferred ARC platform at the time of the training meant that limited hands-on training occurred during that visit, though the tool was delivered shortly thereafter. Natural language incubator display requirements and use cases for its Flight mode were used as the basis for collaboration between the project members with the SCR technology providers. The technology providers encoded some of the requirements as a formal SCR requirements specification; this took about 2 person-weeks. The specification described behavior for setting the chamber fan speed and a goal temperature based on user inputs. The SCR technology providers had planned to give hands-on training on the GUI builder to the project members, but schedule conflicts prevented this. The SCR technology providers provided tutorial materials and remote assistance resulting in the construction of a customized GUI for the incubator display. The project members tested the constructed display GUI against the requirements and found its behavior consistent with the requirements.

The project members reported that no errors in the original natural language requirements and use cases were uncovered in this process, though the SCR technology providers noted there was a lack of completeness and existence of ambiguity in the original natural language requirements that was reflected in the SCR specification. An example of this is

that it was not specified how the functions interacted or conflicted; e.g., what the required behavior should be when a new command is given before the previous command completes.

The project members considered the use of the SCR technology successful in that the SCR requirements specification correctly captured some requirements of the Incubator Display. It does not appear that the project members can develop SCR specifications unaided. The project members recommended, and the technology developers agreed, that the SCR methods and tools should be used when the understanding of the software requirements is mature. The project members concluded that the toolset is valuable for validating requirements prior to design, and made other recommendations regarding extensions to the SCR methods and tools which the technology providers said have been or could easily be implemented.

Success criteria 1, 4 and 7 were met.

## **2.2 IVVF: “Infuse CodeSurfer into NASA Code S IV&V Process”**

CodeSurfer is a commercial tool from GrammaTech, Inc. for browsing C and C++ code. It allows for visualization of data and control flow via, for example, call graphs, and forward and backward slicing. It was employed in a 2004 research infusion collaboration at JSC where it was used during code inspections (see <http://ti.arc.nasa.gov/researchinfusion/collabhistory/2004/2004SARPFinalReport.pdf>).

The goal of the collaboration at IVVF was to apply the tool to analyze flight code for IV&V. The original target software was not available in time, so software for a solar observatory was substituted. The observatory software was about 1.5 MB of C/C++ for C&DH, ACS, and instrument code. A delay was encountered by the tool not being able to ingest this software; this was eventually repaired in a new release of CodeSurfer. The observatory software analysis task was transitioned to another contractor which ended the collaboration. To add value to the collaboration, GrammaTech provided the infusion effort with the results of running its CodeSonar defect detection tool on the observatory software. Because of the various changes in the collaboration, rigorous time and effectiveness comparisons with other tools and previous experience could not be obtained.

CodeSonar identified several defects not identified by other tools or manual analysis. It reported about 60 defects and had a false alarm rate of about 50% which was in line with expectations. The reported defects were analyzed using another tool, Understand for C++ (<http://www.scitools.com/>), for whether they were true defects or false alarms; this took about half an hour per defect, which would have been less if the integrated CodeSurfer/CodeSonar interface (which exists but was not provided) had been used.

The project suggested that the people who will set up CodeSurfer to ingest the target software be familiar with compiler technology, and receive separate training, but users unfamiliar with compiler technology can readily become proficient in using CodeSurfer

once it is set up on the target software and they are familiar with the platform. The project said ease of adoption was enhanced by using the Unix version of CodeSurfer.

The project recommended the continued use of CodeSonar, especially with the integrated interface with CodeSurfer. It also recommended CodeSurfer when the control and data flows are sufficiently complex, and the incurred setup time doesn't swamp the analysis time.

Despite change in prime IV&V contractor, CodeSurfer resides in the IVVF tools lab, and it is being used on another project at the PI contractor.

Success criteria 1, 2, 3, 4, and 7 were met, and 5 is anticipated.

### **2.3 JPL: "Application of SpecTRM at JPL's Advanced Project Design Team (TeamX)"**

SpecTRM is a tool, from Safeware Engineering Corporation, that provides for capture of requirements, assumptions, design, design principles, design rationale, hazards, risks and their linkages.

TeamX is a rapid mission-design team at JPL, with representation from many disciplines; e.g., software, power, science. A TeamX session typically lasts a week or two, and, starting from a mission concept, results in a high-level design (spacecraft design, data return strategy, trajectory, etc.). Historically, rationale for design options and their risks have not been retained, so it is not possible to investigate the sensitivity of the design to changes in the design parameters. The goal of this infusion was to investigate the feasibility and benefit of using SpecTRM to capture the design options considered, the basis for making the design decisions, and the hazards and risks associated with the decisions, and to determine the changes needed to accommodate SpecTRM's use if it were decided to be beneficial.

This infusion used an aerobot mission to Titan as its TeamX test case. The process carried out was for members of TeamX to provide to the technology provider information about their work during the design session, so that the provider could enter the data into SpecTRM. (This was the process as purchase of SpecTRM was not included in the proposal.) The provider organized the information in SpecTRM as system-level goals, requirements, assumptions, constraints, design principles, action items, hazards, and then linkages among them. The data captured was a subset of the information captured in the TeamX directory. The project claimed that the biggest benefit was that these attributes were systematically described and traceable.

The project described conditions and suggestions for the integration of SpecTRM into the TeamX process. For example, it would help to have a knowledge base of previous designs and their rationales. Also, the systems engineer on TeamX should be trained in the tool. Another suggestion is to build an interface to SpecTRM that provides TeamX

members the same format for entering information as they use now; also suggested was a concurrent, multi-user SpecTRM.

A journal paper describing the SpecTRM/TeamX experience is in progress. The technology assistant was hired at JPL, so expertise in SpecTRM will be readily available at JPL.

As far as adopting SpecTRM, the TeamX management will decide what its priorities are and how much funding to allocate to each priority. If it turns out that design rationale capture is a priority and funding is allocated to it, SpecTRM is one of the options TeamX will consider.

Success criteria 1, 4, and 7 were met.

### **3 Paths to further infusion of the technologies**

#### **3.1 SCR**

While SCR's GUI builder seems as though it can be used to design customized user interfaces without too much training, learning how to build SCR specifications is more difficult, depending on the background of the individual. Although the three-day training course introduced the project team to the underlying concepts, the SCR tools, and the overall SCR method, learning to construct an SCR requirements specification, according to the technology providers, requires commitment of solid blocks of time (more than a month) on the part of the project team and is facilitated by interaction with the technology providers and familiarity with mathematical models, such as state machine models. The technology providers suggested that the infusion process should be that they develop an initial SCR specification for a system with which the project team is familiar (about 2 weeks of technology provider time), and then the project team modifies and extends it. The ARC infusion started off that way but did not reach the stage of the project team itself doing the modification and extension of the specification. It is possible that, depending on the background of the project team, this technology in general requires more support and devoted blocks of time to infuse than that made available for this infusion.

#### **3.2 CodeSurfer, CodeSonar**

CodeSurfer seems to exemplify a class of research products whose infusion may best be achieved by supporting a collaboration at each center, so as to allow each center the opportunity to "put its toe in the water". A collaboration at KSC is planned for CY06.

CodeSonar seems comparable and perhaps complementary to other static analysis tools such as Klocwork Inspect and Coverity Prevent; a comparison would be worthwhile.

### 3.3 *SpecTRM*

SpecTRM supports many phases of the lifecycle, e.g. requirements formulation and refinement, design, and safety analysis, and thus adopting it affects many processes in the target organization. One technology selection criterion that the research infusion subgroup uses is to not select technologies that require widespread impact to be effective. But SpecTRM is used heavily by the Japanese space agency. Safeware Corporation earns revenue by providing consulting services about safety, as well as tool sales. The approach of involving someone knowledgeable in the technology in a shadow mode was effective in the JPL infusion in exploring the use of the technology, and in general hiring someone knowledgeable in the technology means that expertise will reside in the organization.

## 4 Technology transfer lessons learned

1. Leading-edge tools sometimes have problems. Technology providers have made efforts to compensate.
2. The profile of effort required to learn new technologies varies with the technology. For example, a few days may be enough to learn a software browsing tool such as CodeSurfer, or to apply SCR tools to an existing SCR specification. But committed blocks of time and more resources and suitable background may be required to become facile with aspects of the technologies, such as SCR specification development, with the expectation that the payoff (such as being able to take advantage of applying the SCR tools) would be worth the effort.
3. Busy researchers and project members may have scheduling conflicts.
4. Sometimes project personnel already have in mind technologies they are interested in and the research infusion effort just provides seed money so that the desired collaboration can take place. This was the case with the JPL/SpecTRM collaboration.
5. NASA is a dynamic environment. It is important to consider the loss of organizational memory as a risk up front and plan for its mitigation. The SSBRP was stopped, and its employees dispersed, so expertise in SCR was dispersed as well. The contractor PI using CodeSurfer lost its prime status so work on analyzing the solar observatory was transferred to a new contractor not necessarily using CodeSurfer; however in this case, the tool remains in the IVVF tools lab and is still in use at the original PI contractor on other projects. Also, project cancellation can be demotivating for learning to use a technology.
6. There are differing answers to the question “What is the next step” – from research infusion to technology transfer. A general solution is unlikely. Some technologies are readily integrated and generalized into a parent organization’s existing processes– they are modifications to existing processes. Various other

- technology-specific approaches may be appropriate.
7. To succeed, training and continued support are needed. For example, “The most successful way to do tech transfer is to put a member of the [technology vendor team] on the development team” – Matt Barry, NASA HQ, (paraphrased). The JPL infusion seems to have adopted that approach.
  8. Overall, Research Infusion’s second set of completed collaborations supports the hypothesis that with selection of appropriate technologies, matching of technology with software development team, and guidance and oversight, infusion of new software engineering technologies can be performed successfully on a minimal budget.

## 5 Acknowledgements

This report incorporates material and suggestions from the following individuals (affiliations are at the time of the collaboration).

### Research Infusion Team

Benedetto Di Vito (LaRC)  
Martin Feather (JPL)  
Michael Hinchey (GSFC)  
Lawrence Markosian, QSS Group Inc. (ARC)

### Project Members and Technology Providers (TP)

Susanne Moran, Intrinsic (ARC)  
Tom Varghese, Intrinsic (ARC)  
Constance Heitmeyer (NRL)(TP)  
Ralph Jeffords (NRL)(TP)

Mike Beims, SAIC (IVVF)  
Stephen Pukansky (IVVF)  
Raju Raymond (IVVF)

Leila Meshkat (JPL)  
Michael Luna (JPL)  
Allen Nikora (JPL)  
Kathryn Weiss (MIT) (TP)  
Nancy Leveson (MIT)(TP)

## 6 Acronyms

ACS: Attitude Control System  
ARC: NASA Ames Research Center  
C&DH: Command and Data Handling  
FSW: Flight Software  
GSFC: NASA Goddard Space Flight Center  
GUI: Graphical User Interface  
HQ: Headquarters  
ISS: International Space Station  
ITAR: International Traffic in Arms Regulations  
IVVF: NASA Independent Verification and Validation Facility  
JPL: NASA Jet Propulsion Laboratory  
JSC: NASA Johnson Space Center  
MIT: Massachusetts Institute of Technology  
NASA: National Aeronautics and Space Administration  
NRL: Naval Research Laboratory  
OSMA: Office of Safety and Mission Assurance  
PF: Penetration Factor  
SARP: Software Assurance Research Program (NASA Office of Safety and Mission Assurance)  
SCR: Software Cost Reduction  
SQA: Software Quality Assurance  
SSBRP: Space Station Biological Research Project  
SWG: Software Working Group  
TP: Technology Provider  
TRL: Technology Readiness Level

## 7 References

### 7.1 Collaboration Reports

“Application of Software Cost Reduction Tools and Methods to On-Orbit Crew Displays Final Report”, Intrinsyx Technologies, NASA ARC, Susanne Moran, ITC-RI-05-1002, October 2005.

“Incubator Display Software Cost Reduction Toolset Software Requirements Specification”, Intrinsyx Technologies, NASA ARC, Susanne Moran, POC, ITC-RI-05-1001, October 2005.

“Memorandum: Applying SCR (Software Cost Reduction) Technology to the Software Requirements Specification of the Incubator Display Software Module”, Constance Heitmeyer and Ralph Jeffords.

“CodeSurfer Technology Infusion Final Report”, SAIC, SAIC-CODESURFER-2, December 9, 2005.

“Application of SpecTRM at JPL’s Advanced Project Design Team (TeamX) Final Report”, Leila Meshkat and Allen Nikora, November 30, 2005.

## **7.2 Technologies**

Technologies deployed in 2005 are described at <http://ti.arc.nasa.gov/researchinfusion/materials/2005/index.php> . There are also the following references.

### **7.2.1 SCR**

See <http://chacs.nrl.navy.mil/personnel/heimmeyer.html>

"Tools for constructing requirements specifications: The SCR toolset at the age of ten," *International Journal of Computer Systems Science and Engineering*, 20(1): 19-35, January 2005.

### **7.2.2 CodeSurfer and CodeSonar**

See <http://www.grammatech.com/> .

### **7.2.3 SpecTRM**

See <http://www.safeware-eng.com/> .