EDCS Rationale Capture Cluster: Executive Overview

Scott R. Tilley
stilley@sei.cmu.edu

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Abstract

This document provides an executive summary of the Rationale Capture cluster of the EDCS project. It outlines the cluster's focus, identifies the primary and secondary participants, and presents a preliminary taxonomy of cluster capability areas. It is expected that this document will change as intra- and inter-cluster collaboration evolves over time.

Keywords: EDCS, rationale capture, cluster

1. Introduction

The DARPA Evolutionary Design of Complex Software (EDCS) project is composed of several *clusters*, each of which are aggregations of related research efforts. Thus document provides an executive summary of the Rationale Capture (RC) cluster. The focus of the cluser is to provide rationale recovery, explaination, and management capabilities by capturing the reasoning behind decisions affecting system evolution.

> **ra•tio•nale** \,ra-shə-'nal\ *n* **1**: an explanation of controlling principles of opinion, belief, practice, or phenomena **2**: an underlying reason: BASIS
>
> **cap•ture** \'kap-chər, -shər\ *n* **1**: **d**: the act of recording in a permanent file <data ~>

The definitions of *rationale* and *capture* given above are from the tenth edition of Webster's Collegiate Dictionary. From a software engineering perspective, rationale refers to the reasoning, thinking, or motivation behind decisions made during a product's lifecycle. For example, why a particular architectural style was chosen during development and what the alternatives were. It is important to capture this rationale in a form that facilitates understanding of the subject system. This information can be used to guide the system's evolution by making explicit the "corporate knowledge" that is usually only implicit and difficult to recapture once lost.

2. Participants

The cluster is composed of eight primary participants and two secondary participants. Primary participants are either exclusively members of the RC cluster, or members of the RC cluster and secondary members of one or more other clusters. Secondary members are those with a primary cluster other than RC, but whose

Participant	Affiliation	Project
Sidney Bailin sbailin@kevol.com	Knowledge Evolution, Inc.	Formal Alternatives Management Integrating Logical Inference and Rationales (FAMILIAR)
Barry Boehm♣ boehm@usc.edu	Computer Science Department, University of Southern California	A Win Win-Based Framework and Toolset for Evolutionary Design of Complex Software (development & packaging)
Balakrishnan Chandrasekaran♣ chandra@cis.ohio-state.edu	Computer and Information Science Department, Ohio State University	Connections Between Software Understanding and Software Composition
Lewis Johnson johnson@isi.edu	Information Systems Institute, University of Southern California	Software Explanation Using Multiple Media (MEDIA-DOC)
Takeo Kaneda♦ tk@cs.cmu.edu	School of Computer Science, Carnegie Mellon University	Technology for Analyzing Change and Composing Reusable, Real-Time Software Components and Applications
K. Narayanaswamy swamy@compsvcs.com	CS3: Computing Services Support Solutions	Software Evolution Through Automatic Monitoring
Daniel Reed reed@cs.uiuc.edu	Department of Computer Science, University of Illinois	Collaboration Support for the Development and Evolution of Complex Systems (Orbit)
Romel Rivera romel@xinotech.com	Xinotech Research, Inc.	Software Evolution by Component Unification
Spencer Rugaber♣ spencer@cc.gatech.edu	College of Computing, Georgia Institute of Technology	Mission Oriented Architectural Legacy Evolution (MORALE)
Olin Shivers♦ shivers@ai.mit.edu	Artificial Intelligence Laboratory, Massachusetts Institute of Technology	A Programming Environment for Evolutionary Software Development (Express)

Table 1: Rationale Capture cluster participants

Participants marked with a ♣ are secondary members of other clusters.
Participants marked with a ♦ are secondary members of the RC cluster.

work is sufficiently related to RC that they are affiliated with the RC cluster as well. It is expected that the cluster topology will evolve over time. Table 1 shows the RC cluster's current participants.

Bailin: The goal of this project is to develop a prototype system, FAMILIAR (Formal Alternatives Management Integrating Logical Inference And Rationales), that builds on the combined capabilities of two database capabilities: Capture and ZD. Capture is a system for managing design alternatives and informal rationales. ZD is a formal system based on functional representations work from the AI field. FAMILIAR supports the full spectrum of rationale: informal to formal, functional and non-functional. It performs automatic inference of functional rationales and supports incremental formalization of informal rationales. It places rationales in the context of design alternatives and tradeoffs. It supports design evolution through feature-based comparison of successive versions.

Boehm: The goal of this project is to provide an emerging capability for capturing critical information on the likely directions of a system's evolution and for incorporating it into the system's architecture and design rationale. Experiments to date with WinWin indicate that a system's stakeholders (users, customers,

developers, maintainers, etc.) identify more desired system capabilities than can be accommodated within initial system budgets and schedules. They negotiate a win-win solution which identifies the top-priority capabilities for initial implementation. In the process, they also establish a prioritized set of desired future capabilities, thus capturing the needed critical information on the likely directions of system evolution.

In addition, this project involves a packaging aspect focused on a set of innovative extensions to the WinWin system to capture critical system evolution information, resulting in new capabilities. Structure for design rationale capture using architecture-based extensions of the WinWin model will be added. An additional set of architecture-based tools for analysis of non-functional attributes, particularly cost, schedule, interoperability, and performance will be provided. A new set of WinWin-related technology evaluation criteria, along with associated tool instrumentation and experimental evaluation approaches will be added. WinWin's model-driver open-architecture support of integrated product teams will be incorporated. Finally, concurrent technology development and experimentation will take place via an existing arrangement of University, Industry, and Government representatives.

Chandra: The major objectives of the project are to develop a discipline and a technology for software composition from components; to develop a formal basis for representing device functions and structure and apply it to software understanding; to investigate issues in reverse engineering; and to develop component libraries in Ada and C++ for use in real software development. Success in this effort will lead to reduced software development costs, increases in the reliability and comprehensibility of software, and advances in reverse engineering and reengineering.

Johnson: Software explanation technology is the dynamic generation of presentations specific to individual users' tasks and needs. It helps reduce the amount of time spent sifting through irrelevant information, helping to ensure that the presented information is more readily understood. Until recently, software explanation technology was focused on generation of (hyper)textual descriptions. The aim of the MEDIA-DOC project is to develop software explanations that integrate diagrams, animations, and text. Extends work from the I-DOC Dynamic Documentation project at ISI/USC. The project will develop new graphical presentation formats (and tools for viewing and editing descriptions in these formats), techniques for extracting from design repositories information necessary for explanation, and techniques for conveying specific information via graphical media. These tools will be built around commonly-available presentation technologies and instrumented so that usage data can be collected during system operation for subsequent empirical evaluation

Kaneda: The goal of this project is to provide a way to reason and capture knowledge about anticipated changes (change rationale), an analytical method for designing software components that minimize the impact of future changes to the systems in which they are embedded (design rationale), an analytical method for extracting semantic (algorithmic) information from existing software (reengineering rationale), and a virtual software design environment that provides an open and multi-modal software infrastructure for the transparent access of physically distributed software and hardware resources. The overarching focus of the project is to propose and prove feasible innovative technologies that reduce the cost and time to build high-quality real-time software applications. The project will investigate methods to capture knowledge about change and will propose a meta-level process for creating a taxonomy of changes that is germane to a particular domain or class of applications.

Narayanaswamy: The goal of this project is to improve dramatically the adaptability of software systems that are required to operate reliably in ever-changing environments. This is achieved by monitoring a complex system's run-time state and environment for satisfaction of its design and installation requirements and assumptions. It seeks a uniform monitoring foundation over the entire software lifecycle to make this possible. System characteristics are articulated as they become known at different points in its lifecycle in a language designed for that purpose. From these formal statements, run-time monitors are automatically generated and installed either as part of or alongside the running system as applicable. Monitors detect

violations of assumptions, but also have a response component that can be automatically enacted if metaprogramming/process support are available in the underlying software development environment.

Reed: The goal of this project is to create a prototype extensible, evolving collaboration framework called Orbit. It will support the full range of activities undertaken by current and future team members, integrate the wide variety of tools and artifacts they use to realize their goals, and facilitate the evolution of the development team and its processes over time. Extends previous work on wOrlds. Provides an open environment and infrastructure for collaboration support through the notion of locales (virtual settings for interaction) using the WWW and ORB-based software tools. Adaptive user interfaces and Internet scalability will also be addressed.

Rivera: The goal of this project is to bring a new paradigm of support for the evolution of legacy applications. This will include understanding legacy applications at a higher abstraction and transforming them into complete inter-relating object-oriented subsystems, devising strategies so that plans can be automatically deduced from existing software, and devising strategies so that program transformation rules can be automatically generated through plan comparision. The work involves the continued development of the Ada Object Abstractor to support new transformations and interactive visual support to parallelize Ada 95 applications using XPAL (Xinotech Pattern Abstraction Meta-Language).

Rugaber: Mission Oriented Architectural Legacy Evolution (MORALE). Three components. (1) consists of a mission-directed requirements determination process that, through a systematic process of inquiry and refinement, turns mission-oriented goals into specifications of the desired behaviors of architectural components. Extends previous work on Inquiry-Based Requirements Analysis and the use of schematic scenarios to understand requirements. (2) An architectural assessment process that can ascertain the impact of new requirements on an existing system's architecture. Focus is on the analysis of architectural descriptions in order to determine how well they satisfy a large class of quality criteria. Extends previous work on Software Architecture Analysis Method to perform this analysis. (3) A reverse engineering process for extracting architectural information from an existing system. Beyond the traditional structural analyses provided by existing commercial and research tools, MORALE will extract behavior information from dynamic analysis of the message flow among architectural components. Extends previous work on cliché recognition (detection of commonly occurring patterns in code) and interleaving detection (determination of situations in which code implements more than one design goal or requirement).

Shivers: The goal of this project is to develop a system called Express that aids software understanding. In particular, a design rationale management and reasoning system based on the use of natural-language annotations, a programming system that allows for run-time metering to be fed back into the compiler's optimiser and code generator, and a programming environment that allows programmer-supplied formal annotations to be exploited by the compiler and run-time system for generating optimised safe code. The work draws upon AI and programming language research.

3. Taxonomy

Although the individual participants within the cluster are working on different aspects of RC, commonalties are apparent. A preliminary taxonomy of capability focus areas is shown in Table 2. Other possible categories include analysis and maintenance. As with the makeup of the cluster itself, it is expected that this taxonomy will change over time.

The current categories were chosen to reflect the lifecycle focus of the rationale capture capability. The "recovery" capability focuses on post-deployment extraction of rationale from an existing system, using techniques such as domain analysis and reverse engineering. The "explanation" capability focuses on aiding software and system understanding. The "management" capability focuses on monitoring and maintaining assumptions, capabilities, or constraints at various levels of abstraction as the system evolves over time.

Capability Focus Area	Participants
Recovery	Chandra, Kaneda, Narayanaswamy, Rivera, Rugaber, Shivers
Explanation	Boehm, Johnson, Narayanaswamy, Shivers
Management	Bailin, Boehm, Kaneda, Reed

Table 2: Preliminary RC taxonomy

Acknowledgments

This work was supported in part by the U.S. Department of Defense.