STARS Framework for Reuse Processes

Margaret J. Davis

Boeing Defense & Space Group
P.O. Box 3999 MS 87-37
Seattle, Washington 98124-2499
mjdavis@stars.boeing.com

Abstract

Incorporation of reuse concepts into software-intensive systems development so that reuse becomes systematic requires guidance as to what constitutes reuse-based activities. The reuse process framework discussed in this article provides a common viewpoint for considering, defining, composing, and applying reuse-based activities. The framework is based on the notion that discrete process units can be well-defined and can be composed into broader, consistent process contexts such as life cycle models. The framework is asset-centered and is tailorable to the needs of specific organizations, specific domains, and specific projects. The framework is also independent of any life cycle model style such as waterfall or spiral.

Keywords: reuse processes, reusable assets, reuse-based development

1 Introduction

The Software Technology for Adaptable, Reliable Systems (STARS) program has articulated a mission to accelerate the shift to a *"process-driven, domain-specific reuse-based, technology-supported"* development paradigm for software-intensive DOD systems. In support of this mission, the three STARS' prime contractors (Boeing, IBM, and Unisys) are jointly developing a description of the reuse aspects of such a software development paradigm. This description is captured in an evolving document called the STARS Reuse Concept of Operations. The focus of version 0.5 [1], a recently-released first draft, is a framework for considering and defining reuse supporting process building blocks and for composing these building blocks into broader contexts such as reuse-based organizational strategies and product life cycles. The remainder of this paper gives an overview of that reuse process framework.

2 Reuse Process Framework Overview

STARS has identified functions and processes supporting reuse in the context of software-intensive system development and maintenance. Further, these reuse supporting activities have been organized into a process framework containing four families of processes. The names of these families emphasize the primary purpose of each. The reuse process families (see Figure 1) are:

- reuse planning;
- asset creation;
- asset management; and,
- asset utilization.

The families of the reuse process framework can be decomposed further to identify processes and functions focusing on different aspects of each family's purpose. Individual organizations may use different decompositions of these families to suit their goals and business strategies. Brief descriptions of processes in the decomposition currently being used by STARS are given in following sections 2.1, 2.2, 2.3, and 2.4.

The arrows in Figure 1 represent the extensive information flow, influence, and feedback among the four process families. In general, the arrows represent the flow of decisions, constraints, experience lessons, and assets.

As the figure shows, inputs to the reuse process framework are market forces, existing assets, systems, domain expertise, and tools. A market force is defined as requirements or needs of any intended customer. Software systems, software architectures, software components, asset libraries, experience reports, domain analysis results, and domain models are examples of the output software and related products shown in the figure.

The results of the *asset planning* processes feed separately into the *asset creation*, *asset management*, and *asset utilization* process families. Planning processes set goals and strategies, select and effect the tailoring of processes consistent with the goals and strategies, and identify and allocate existing resources. The *asset creation* process family produces software and software related assets. The *asset management* process family evaluates, describes, and organizes the assets provided by the *asset creation* process family. The *asset utilization* process family accesses the organized assets to construct software-intensive systems.

Lessons learned regarding the usage, applicability, quality, and reusability of assets are feedback from the *asset utilization* processes to the *asset management* processes. Lessons learned regarding missing assets or possible asset generalizations are feedback from the *asset utilization* processes into the *asset creation* processes. Lessons learned regarding asset quality and description are feedback from the *asset management* processes to the *asset creation* processes. Needs for new assets; lessons learned regarding process usage, applicability, and quality; and new process assets are feedback from the *asset creation*, *asset management*, and *asset utilization* processes into the *asset planning* processes.

Figure 1: STARS Reuse Process Framework

2.1 Reuse Planning

An important function of the planning activity in Figure 1 is to define a reuse strategy and plan for its implementation within the organization that is undertaking a reuse program. A second function is to implement the strategy in plans and processes for a specific project. A related function is to measure and evolve the process for executing the plans. Note that many of the planning activities and products are appropriate at both the organizational and specific project levels.

Reuse Strategy Development A reuse strategy is used to guide the asset creation, management, and utilization processes. The activities required to define the strategy will depend on the nature of the organization, e.g., whether it is a company seeking to market reusable components or develop systems based on them, a DoD Program Executive Officer establishing a reuse program for a given domain, a Program Manager developing a specific system, or a maintenance organization. The strategy will be influenced by the organization's goals and top level reuse policy. The reuse strategy may define processes that identify, evaluate and select domains for reuse; define a set of methods for asset creation that are compatible with the methods for asset utilization; create plans for asset creation, management, and utilization; and define goals to measure the effectiveness of reuse. A software reuse strategy may include, but is not limited to, a domain selection method [2, 3], an asset creation plan, an asset management plan, an asset utilization plan, and process and product improvement plans.

Incorporation of Reuse Into the Project Process If the parent organization of a specific reuse project has produced generic plans for asset creation, utilization, management, and process improvement, then incorporation of reuse into the project process is a matter of adapting those generic plans to the project's particular situation. If no generic plans exist, specific plans for asset creation, utilization, management, and process improvement will need to be developed. In either case detailed project specific plans will result.

Process Measurement and Evolution The reuse process measurement and evolution function receives input in the form of data captured about the asset creation, management, and utilization processes and products. It also receives lessons learned, asset requirements, process requirements, and any other form of relevant feedback from individuals involved in those processes. Feedback from the users of the software products is also input to this function. The process involves analysis of the input information; identification of problems and opportunities for improvement; development of solutions; identification of resources required to effect the solution; definition of changes to the process or products; modification of the plans and the process being followed; and measurement and analysis of the modified process.

2.2 Asset Creation

Since *asset* is a very broad term, the activities identified as asset creation include domain analysis, domain modeling, software architecture/design development, reverse engineering, design recovery, software component development, application generator development, and source code translation. Furthermore, the encoded domain expertise and design rationale guide the creation of software components, architectures, designs, and application generators. That is, there is considerable interaction and feedback among the members of the *asset creation* process family.

Domain Analysis and Modeling The goal of domain analysis is to develop a domain model, reusable requirements, and domain variability description applicable to solution systems within the domain. Note that domain is being used here in its broadest sense, i.e, as an area of activity or knowledge. At a high level, domain analysis is a combination of reverse engineering, knowledge extraction, technology and requirements forecasting, and modeling.

Software Architecture Development The purpose of this activity is to produce an architecture that can be used to implement numerous systems for the domain as defined by the domain analysis. The process of architecture development seeks to identify a set of software components and their interactions that can support both the full and minimal set of domain services and objects [4].

Software Component Development The goal of this activity is to develop reusable software components that implement the previously developed domain-specific architecture. Before this activity is undertaken, reuse planning has already evaluated whether component development is more appropriate than or complementary to application generator development or use. Reuse planning activities will also have evaluated whether translation of code from legacy systems may also be appropriate.

Application Generator Development The goal of application generator development is to provide a capability that allows a reuser or application developer to create software (sub)systems using the concepts and terms belonging to the domain. The point is to support the end user in stating "what" is desired rather than detailing "how" the desired effect is to be achieved. This "what" orientation can also be termed requirements-based.

Asset Evolution The results of asset evaluations from the *asset management* and *asset utilization* process families are feedback into the *asset creation* processes. There should be explicit processes that receive and analyze these results with the objective to enhance the appropriate domain model, software architecture and components, and application generators. The feedback may also be used to improve or better tailor the processes of modeling, component and architecture creation, and application generator development to the needs of particular domains or organizations.

2.3 Asset Management

The goal of asset management is to acquire, evaluate, describe, and organize reusable assets to assure their availability to asset creation and asset utilization processes. Asset management is also responsible for asset library administration and operation. Asset management activities include asset acquisition, asset acceptance, asset classification, asset cataloging, asset certification, library and asset metrics collection, library administration and operation, and asset maintenance and enhancement.

Asset Acquisition The goal of asset acquisition is to obtain assets from external asset libraries and other sources in support of asset creation and asset utilization activities.

Asset Acceptance The goal of asset acceptance is to ensure that an asset satisfies all legal and policy constraints and that sufficient information is available to catalog the asset.

Asset Classification The goal of asset classification is to develop a scheme for categorizing assets on the basis of their domain-relevant characteristics. The classification scheme provides library users with an organizational framework for locating and understanding domain assets. This is distinct from the process (described under asset cataloging below) of determining where a particular asset belongs within the library classification scheme.

Asset Cataloging Asset cataloging is broken down into three steps: asset categorization, asset description, and asset installation. Asset categorization (as distinguished from the asset classification process discussed above) is the process of determining where an asset belongs within the classification scheme. Asset description is the process of creating, capturing, or adapting all the information that is needed to describe the asset in the context of the library's data model, once the asset has been categorized. Asset installation is the process of installing the categorized and described asset in the library system.

Asset Certification The ultimate goal of asset certification is to guarantee that software assets implement their requirements and that their execution will be error free in their intended environment.

Library and Asset Metrics Collection Library metrics are used to measure the effectiveness of library management processes, tools, and policies. Asset metrics are used to measure the characteristics and effectiveness of individual assets, such as their reusability. The goal of collecting such measurements is to improve the effectiveness of the library in supporting reuse processes within client organizations.

Library Administration and Operation The goal of library administration and operation is to assure the availability of the asset library for asset creation and asset utilization activities.

Asset Maintenance and Enhancement The goal of the asset maintenance and enhancement process is to iteratively improve the assets in the library relative to user and domain needs.

2.4 Asset Utilization

There are two primary methods of asset utilization, corresponding to system composition and system generation. These two asset utilization methods are complementary and can both be employed within the same domain or for a single system development. The other asset utilization processes (asset identification, asset understanding/evaluation/selection, and asset tailoring/integration) are subordinate to the two primary asset utilization methods and are approached differently within each method.

System Composition Asset-based system composition is a process in which the software engineer constructs new products (e.g., requirements, design, code, tests, documentation) from previously developed or newly generated parts. This is typically done by identifying, understanding, evaluating, and selecting appropriate generalized domain assets and tailoring and integrating them to meet specific system needs.

System Generation System generation is a process for producing systems or subsystems that ideally incorporates all the variation in a domain into a set of parameters expressed in terms of a specification language or template. A generation tool accepts specifications from engineers that define values for the domain parameters and resolves the variation accordingly to generate components of the target system. The specifications are generally non-procedural in nature and can be expressed in a number of different forms (e.g., textual, graphical, template-based, etc.). These specifications in effect define a set of specific target system requirements that lie within a set of more generic domain requirements represented by the specification language. Thus, the target components are derived directly from a specification of system requirements, which is why generation is often referred to as requirements-based reuse.

Feedback to Reuse Planning, Asset Creation, and Asset Management For each reuse-based development effort, assets within the relevant domain(s) will likely need to be updated based on feedback from asset utilization. If updates are appropriate, the domain model may be amended, new assets may be constructed, and other assets may be changed or deleted. Similarly, each reuse-based development effort should yield lessons that can be applied to asset management within the domain. Engineers' experiences with browsing and querying the library may result in recommendations for refining or correcting aspects of the library taxonomy or asset descriptions; experiences with the tools used to facilitate asset understanding, tailoring, integration, and generation may yield recommendations for additional tools or improvements to the existing tools; problems with assets that were thought to be well-qualified may reveal inadequacies in the asset qualification process; lack of adequate access to the remote libraries may result in recommendations for improved library connectivity or interoperability.

3 Using the STARS Reuse Process Framework

Historically, organizations have based their software development plans on methodology, technique, or tool selections made to implement an idealized project life cycle rather than on process building block selections. Indeed, software development has mostly been considered as one gigantic waterfall life cycle divided into major phases encompassing system conception to demise. In contrast, STARS is promoting the concept that there are multiple, valid modern software life cycle models appropriate for different organizational goals, strategies, and strengths. That is, STARS is generalizing the concept of life cycle model from a strategy for software *system* development to strategies for software *product* development, where product includes components, interface and protocol standards, architectures, domain models, application generators, and systems.

As opposed to modeling and planning a development strategy around major activities and tools, the reuse process framework supports the notion of composing a life cycle model from process building blocks. We believe the benefits of this approach to be:

- The ability to compose multiple, reuse-based life cycle models with different goals guided by the reuse process framework.
- Easier implementation and tailoring of life cycle models in support of individual domains, organizations, and engineers.
- Easier management, measurement, monitoring, and improvement, of life cycle model implementations and improvement in life cycle models.
- Identification of the similarities in appropriate methods, techniques, and tools supporting various life cycle models and processes.

These benefits accrue because process building blocks are well-defined, may have formal representations, have definite begin and end points, have definite start and stop criteria, span a shorter time duration than life cycle phases, and can be customized to available tools and environment support.

References

[1] STARS. STARS Reuse Concept of Operations, Version 0.5, August 1991. Unisys STARS CDRL GR-7670-(NP).

[2] Robert R. Holibaugh, Sholom G. Cohen, Kyo C. Kang, and Spencer Peterson. Reuse: Where to Begin and Why. In *Proceedings of Tri-Ada '89*, pages 266–277, New York, NY, October 1989. Association for Computing Machinery.

[3] A. Jaworshi, F. Hills, T. Durek, S. Faulk, and J. Gaffney. A Domain Analysis Process. Technical Report DOMAIN_ANALYSIS-90001-N, Software Productivity Consortium, SPC Building, 2214 Rock Hill Road, Herndon VA, 22070, January 1990.

[4] D. Parnas. Designing Software for Ease of Extension and Contraction. *IEEE Transactions on Software Engineering*, SE-5(2):128–138, March 1979.

4 About the Author

Ms. Margaret J. Davis is the technical lead for reuse on the Boeing STARS project, which is sponsored in part by DARPA under USAF contract F19628-88-D-0028. Ms. Davis has served in a lead capacity on the Boeing STARS project since its award in 1988. Prior to the STARS project, Ms. Davis was a technical contributor to the Software GetPRICE program, which produced an expert systems tool supporting systems integration testing, and to the Air Force contract that developed a Specification Technologies Handbook. Ms. Davis has 18 years experience in software engineering and received an M.S. in Computer and Information Sciences from the University of Delaware in 1984.