Software Engineering Institute Documents

Maintained by SEI Technical Communication

September 2000

Carnegie Mellon
Software Engineering Institute

Pittsburgh, PA 15213-3890

Software Engineering Institute Documents

Maintained by SEI Technical Communication

September 2000

This report was prepared for the

SEI Joint Program Office
HQ ESC/AXS
5 Eglin Street
Hanscom AFB, MA 01731-2116

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

FOR THE COMMANDER

Norton L. Compton, Lt Col., USAF
SEI Joint Program Office

Contents

Accessing SEI Documents

Online Copies

Each SEI report lists a URL that contains download information for the corresponding report. Please visit the SEI Web site at the provided URL and forward any questions using the *Contact Us* button that appears on the SEI page.

Paper Copies

Paper copies of the reports listed in this document are available from the following organizations.

NTIS (National Technical Information Service)
U.S. Department of Commerce
Springfield, VA 22161-2103
Phone: 703 / 487-4600

DTIC (Defense Technical Information Center)
8725 John J. Kingman Road, Suite 0944
Ft. Belvoir, VA 22060-6218
Phone (toll free in the U.S.): 1-800-225-3842
or 703 / 767-8222

(Note: To order documents from DTIC, you must provide the ADA/DTIC accession number for the desired report.)

2000 Reports

Technical Reports

CMU/SEI-2000-TR-011

ARC, V1.0 Assessment Requirements for CMMI℠, Version 1.0

CMMI Product Development Team

The Assessment Requirements for CMMI (ARC) V1.0 defines the requirements considered essential to assessment methods intended for use with CMMI models. In addition, a set of assessment classes is defined based on assessment usage scenarios. These classes are intended primarily for developers of assessment methods to use with CMMI capability models in the context of the CMMI Product Suite. Additional audiences for the document include lead assessors, and other individuals who are involved in or may be interested in process assessment or improvement.

The approach employed to provide guidance to assessment method developers is to define a class of assessment method usage scenarios (which are based on years of experience in the process improvement community) called assessment classes. Requirements are then allocated to each class as appropriate based on the attributes associated with that class. Thus, a particular assessment method may declare itself to be an ARC class A, B, or C assessment method. This designation implies the sets of ARC requirements which the method developer has considered when designing the method.

Assessment methods which satisfy all of the ARC requirements are called class A methods; in addition to being used to render ratings for benchmarking purposes, class A assessment methods can be used to conduct 15504-conformant assessments.

More information on the CMMI product suite is available on the World Wide Web at http://www.sei.cmu.edu/cmmi.

http://www.sei.cmu.edu/publications/documents/00.reports/00tr011.html

CMU/SEI-2000-TR-008
ADA379930

Volume II: Technical Concepts of Component-Based Software Engineering

Bachman, F.; Bass, L.; Buhman, C.; Comella-Dorda, S.; Long, F.; Robert, J.; Seacord, R.; Wallnau, K.

The Software Engineering Institute is undertaking a feasibility study of “component-based software engineering” (CBSE). The objective of this study is to determine whether CBSE has the potential to advance the state of software engineering practice and, if so, whether the SEI can contribute to this advancement. This report is the second part of a three-part report on the study. Volume I contains a market assessment for CBSE. Volume III outlines a proposed course of action for the SEI. Volume II, this report, establishes the technical foundation for SEI work in CBSE. The paper asserts that the key technical challenge facing CBSE is to ensure that the properties of a system of components can be predicted from the properties of the components themselves. The key technical concepts of CBSE that are needed to support this vision are described: *component*, *interface*, *contract*, *component model*, *component framework*, *composition*, and *certification*.

http://www.sei.cmu.edu/publications/documents/00.reports/00tr008.html

CMU/SEI-2000-TR-005
ADA377438

Analysis of Lead Assessor Feedback for CBA IPI Assessments Conducted July 1998-October 1999

Dunaway, D.; Seow, M.; Baker, M.

In the Appraiser Program of the Software Engineering Institute (SEI), authorized Lead Assessors lead Capability Maturity Model-Based Appraisals for Internal Process Improvement (CBA IPI). At the conclusion of each assessment, they are required to submit certain artifacts to the SEI. Data from assessments is recorded to provide

the community with information on the state of the software community's process maturity, as related to the Capability Maturity Model® (CMM®) for Software Version 1.1. These data can be viewed on the SEI Web site: http://www.sei.cmu.edu/sema/profile.html.

Additional feedback data are required of a Lead Assessor in order to monitor the consistency of use of the assessment method for quality control purposes. Data are collected from Lead Assessors, assessment team members, and sponsors of the assessments. The results reported in this document reflect information sent to the SEI by Lead Assessors through a Lead Assessor's Requirements Checklist. The checklist aids the Lead Assessors in keeping track of their implementation of each of the method's requirements. The checklist also provides information back to the community regarding metrics being reported by Lead Assessors; this helps in more effective planning for future assessments. In addition, the checklist acts as a quality control mechanism to monitor the consistency of use of each of the method's activities.

Thanks to the Lead Assessors who contributed the data in order that it can be shared with other Lead Assessors and the community.

http://www.sei.cmu.edu/publications/documents/00.reports/00tr005.html

CMU/SEI-2000-TR-004 *ATAM: Method for Architecture Evaluation*

Kazman, R.; Klein, M.; Clements, P.

If a software architecture is a key business asset for an organization, then architectural analysis must also be a key practice for that organization. Why? Because architectures are complex and involve many design tradeoffs. Without undertaking a formal analysis process, the organization cannot ensure that the architectural decisions made—particularly those which affect the achievement of quality attribute such as performance, availability, security, and modifiability—are advisable ones that appropriately mitigate risks. In this report, we will discuss some of the technical and organizational foundations for performing architectural analysis, and will present the Architecture Tradeoff Analysis Method[SM] (ATAM[SM]) a technique for analyzing software architectures that we have developed and refined in practice over the past three years.

http://www.sei.cmu.edu/publications/documents/00.reports/00tr004.html

CMU/SEI-2000-TR-003 *Improving the Acquisition of Software Intensive Systems*

Goldenson, D.; Fisher, M.

Acquisitions of software intensive systems by the Department of Defense (DoD) have often suffered from poor product quality, cost overruns, and schedule slips. In turn, these problems have frequently been linked to the inability of project offices to successfully manage the acquisition of the software components of the systems.

There have been a number of efforts to provide the necessary education and training to improve the skills and capabilities of managers for software intensive acquisitions. However, acquisition problems remain pervasive in the DoD.

More must be known about the causes and underlying issues surrounding these problems. Specifically, the needs of the acquisition management offices must be better understood to help them improve. This includes a better understanding of how education and training can improve the individual manager's skills and competency related to acquiring such systems.

To elicit these needs, the Software Engineering Institute (SEI) conducted a survey of senior acquisition managers. The survey focused on the performance of their organizations, particularly with respect to a series of skills and competency areas that may affect an organization's ability to successfully acquire software intensive systems.

Results indicate that the program executive officers (PEOs) and program managers (PMs) who completed the survey were reasonably well satisfied with the capabilities of their organizations to acquire software intensive systems. In many cases, however, the source of the expertise for such acquisitions were contractors either supporting the organizations or the prime contractors developing these systems. Comparable expertise often was unavailable in government acquisition organizations themselves. From this fact, the need for government expertise in these acquisitions was noted. In addition, the survey queried participants on the best way to obtain this expertise through education and training.

Finally, recommendations derived from survey results are offered to increase software acquisition education and training opportunities for managers.

http://www.sei.cmu.edu/publications/documents/00.reports/00tr003.html

CMU/SEI-2000-TR-002
ADA375843

Fourth Product Line Practice Workshop Report

Bass, L.; Clements, P.; Donohoe, P.; McGregor, J.; Northrop, L.

The Fourth Software Engineering Institute (SEI) Product Line Practice Workshop was a hands-on meeting held in December 1999 to share industry practices in the area of tool support for software product lines, to explore the technical and non-technical issues involved, and to evolve the SEI Product Line Practice Framework. This report synthesizes the workshop presentations and discussions, which described practices and issues associated with tool support for software product lines.

http://www.sei.cmu.edu/publications/documents/00.reports/00tr002.html

CMU/SEI-2000-TR-001
ADA375851

Architecture Based Design Method, The

Bachmann, F.; Bass, L.; Chastek, G.; Donohoe, P.; Peruzzi, F.

This paper presents the Architecture Based Design (ABD) method for designing the high-level software architecture for a product line or long-lived system. Designing an architecture for a product line or long-lived system is difficult because detailed requirements are not known in advance. The ABD method fulfills functional, quality, and business requirements at a level of abstraction that allows for the necessary variation when producing specific products. Its application relies on an understanding of the architectural mechanisms used to achieve this fulfillment.

The method provides a series of steps for designing the conceptual software architecture. The conceptual software architecture provides organization of function, identification of synchronization points for independent threads of control, and allocation of function to processors. The method ends when commitments to classes, processes and operating system threads begin to be made. In addition, one output of the method is a collection of software templates that constrain the implementation of components of different types. The software templates include a description of how components interact with shared services and also include “citizenship” responsibilities for components.

http://www.sei.cmu.edu/publications/documents/00.reports/00tr001.html

Technical Notes

CMU/SEI-2000-TN-008

Mining Existing Assets for Software Product Lines

Bergey, J.; O’Brian, L.; Smith, D.

Mining of existing assets offers an organization the potential to leverage all, or part, of its cumulative system investments, and thus represents a critical practice area in implementing a software product line. However, there are significant risks in achieving success because of the poorly documented and maintained state of many existing systems and the fact that many systems were initially developed for different paradigms than current distributed, Web-oriented, object-oriented approaches.

Four basic steps are required to successfully mine assets: 1) preliminary information gathering, 2) making decisions on whether to mine assets and which type of overall strategy to use, 3) obtaining detailed technical understanding of existing software assets, and 4) rehabilitation of assets.

This note outlines basic considerations for each of these steps. It outlines typical information to collect before an analysis. It then outlines a model for making decisions on mining legacy assets, and discusses the technical understanding of assets and the rehabilitation of assets.

Because of its importance as a strategy for product lines, architecture reconstruction is discussed, as it is supported by an automated tool set known as the Dali workbench.

http://www.sei.cmu.edu/publications/documents/00.reports/00tn008.html

CMU/SEI-2000-TN-004
ADA377385

Guidelines for Using OAR Concepts in a DoD Product Line Acquisition Environment

Bergey, J.; Smith, D.

Many DoD organizations are considering product line initiatives as a means of overcoming the issues of quality, cost, and schedule inherent in a "one-at-a-time" system development or acquisition paradigm. Because a product line approach revolves around the creation of a comprehensive set of core assets, mining and adapting (i.e., reengineering) existing legacy system assets can offer significant leverage. By mining and adapting these assets, an organization can exploit the proven capabilities of existing systems, reduce the funding required, and develop/ acquire new systems of higher quality within a shorter time frame.

This technical note focuses on providing guidance for DoD organizations for mining legacy systems to obtain core assets that will fit into a previously defined software architecture for a product line. We explain how insights from a conceptual model, Options Analysis for Reengineering (OAR), can be used in an acquisition context to provide the government with an approach for obtaining greater insight and understanding into a contractor's proposed technical reengineering approach.

http://www.sei.cmu.edu/publications/documents/00.reports/00tn004.html

CMU/SEI-2000-TN-003
ADA377453

Survey of Legacy System Modernization Approaches, A

Comella-Dorda, S.; Wallnau, K.; Seacord, R.; Robert, J.

Information systems are critical assets for modern enterprises and incorporate key knowledge acquired over the life of an organization. Although these systems must be updated continuously to reflect evolving business practices, repeated modification has a cumulative effect on system complexity, and the rapid evolution of technology quickly renders existing technologies obsolete. Eventually, the existing information systems become too fragile to modify and too important to discard. However, organizations must consider modernizing these legacy systems to remain viable. The commercial market provides a variety of solutions to this increasingly common problem of legacy system modernization. However, understanding the strengths and weaknesses of each modernization technique is paramount to select the correct solution and the overall success of a modernization effort. This paper provides a survey of modernization techniques including screen scraping, database gateway, XML integration, database replication, CGI integration, object-oriented wrapping, and "componentization" of legacy systems. This general overview enables engineers performing legacy system modernization to preselect a subset of applicable modernization techniques for further evaluation.

http://www.sei.cmu.edu/publications/documents/00.reports/00tn003.html

CMU/SEI-2000-TN-002
ADA377656

Modeling the Space Shuttle Liquid Hydrogen Subsystem

Atanacio, B.

This paper describes experiences with modeling the liquid hydrogen subsystem of the space shuttle. The Symbolic Model Verifier tool and the Software Cost Reduction tool set were used to model and specify the behavior of the system. The tools were then used to check for errors in the models. Modeling a problem from several different perspectives offers the chance to uncover discrepancies among different models and to understand the problem space enough to ask important questions about the behavior of the system. Each tool presented different issues in modeling the problem. Both models and a breakdown of the time spent during this study are included as appendices.

http://www.sei.cmu.edu/publications/documents/00.reports/00tn002.html

CMU/SEI-2000-TN-001
ADA375859

Basic Concepts of Product Line Practice for the DoD

Bergey, J.; Fisher, M.; Gallagher, B.; Jones, L.; Northrop, L.

Industrial experience demonstrates clearly that a product line approach for software-intensive systems can save money and result in faster time to field higher quality systems. Many within the Department of Defense (DoD) recognize the benefits of product lines, but also recognize that there are significant challenges to adopting this approach. Many of these challenges stem from the fact that the DoD is in the business of acquiring systems rather than developing them.

The Product Line Systems Program is publishing a series of technical notes designed to condense knowledge about product line acquisition practices into a concise and usable form for the DoD acquisition manager and practitioner.

This technical note provides background information about product lines to serve as a foundation for other technical notes in this series. Key terms, concepts, and benefits of a product line approach are given. Additionally, concepts of product line acquisition in a DoD context are discussed.

http://www.sei.cmu.edu/publications/documents/00.reports/00tn001.html

Special Reports

CMU/SEI-2000-SR-008

Spiral Development: Experience, Principles, and Refinements
Spiral Development Workshop February 9, 2000

Boehm, B. (edited by W.J. Hansen)

Spiral development is a family of software development processes characterized by repeatedly iterating a set of elemental development processes and managing risk so it is actively being reduced. This paper characterizes spiral development by enumerating a few "invariant" properties that any such process must exhibit. For each, a set of "variants" is also presented, demonstrating a range of process definitions in the spiral development family. Each invariant excludes one or more "hazardous spiral look-alike" models, which are also outlined. This report also shows how the spiral model can be used for a more cost-effective incremental commitment of funds, via an analogy of the spiral model to stud poker. An important and relatively recent innovation to the spiral model has been the introduction of anchor point milestones. The latter part of the paper describes and discusses these.

http://www.sei.cmu.edu/publications/documents/00.reports/00sr008.html

CMU/SEI-2000-SR-006

Spiral Development—Building the Culture
A Report on the CSE-SEI Workshop, February, 2000

Hansen, W.J.; Foreman, J.T.; Carney, D.J.; Forrester, E.C.; Graettinger, C.P.; Peterson, W.C.; Place, P.R.

A number of organizations are successfully applying the Spiral Development Model (SDM) and finding it valuable in addressing such challenges as rapid development, COTS (commercial-off-the-shelf) software integration, new technologies, and product line management. However, other organizations have experienced difficulties with spiral development—due to over-relaxed controls, underestimated risks, existing sequential development policies, inflexible financing mechanisms, ingrained cultures, and confusion about what spiral development is and how to apply it. To attack these problems, a workshop was held February 9-11, 2000, at the University of Southern California under the sponsorship of its Center for Software Engineering (CSE) and the Software Engineering Institute (SEI) of Carnegie Mellon University. Work groups at the workshop recommended specific actions aimed at building and spreading a culture for the SDM community. These can be described as defining, improving, promoting, and studying SDM, educating about SDM, adapting to SDM, and enhancing teamwork. This report summarizes the workshop and presents its recommendations.

http://www.sei.cmu.edu/publications/documents/00.reports/00sr006.html

CMU/SEI-2000-SR-004
ADA377988

Software Architecture Documentation in Practice: Documenting Architectural Layers

Bachmann, F.; Bass, L.; Carriere, J.; Clements, P.; Garlan, D.; Ivers, J.; Nord, R.; Little, R.

This report represents the first milestone of a work in progress. That work is a comprehensive handbook on how to produce high-quality documentation for software architectures. The handbook, tentatively entitled Software Architecture Documentation in Practice, will be published in mid- to late-2000 by Addison Wesley Longman as a book in the SEI series on software engineering. Aimed squarely at the practitioner, the handbook is intended to fill a gap in the literature: There is a complete lack of language-independent guidance about how to actually capture an architecture in written form so that it can fulfill its purpose as a communication vehicle providing a unified design vision to all of the varied stakeholders of a development project.

The theme of the work is that documenting an architecture entails documenting the set of relevant views of that architecture, and then completing the picture with documentation of information that transcends any single view. The report lays out our approach and organization for the complete book, and provides full guidance for one of the most commonly used architectural views: the layer diagram. The audience for this book is the community of practicing architects, apprentice architects, and developers who are on the receiving end of architectural documentation.

http://www.sei.cmu.edu/publications/documents/00.reports/00sr004.html

CMU/SEI-2000-SR-003
ADA377440

November 1999 High Maturity Workshop, The

Paulk, M.; Chrissis, M.B.

A workshop for high maturity organizations was held on November 16-18, 1999, at the Soft-ware Engineering Institute (SEI) in Pittsburgh. The purpose of this workshop was to better understand practices that characterize Level 4 and 5 organizations. Topics of discussion included both practices described in the CMM (Capability Maturity Model) and other practices that have a significant impact in mature organizations. Two themes were anticipated to be important to the workshop participants: statistical process control for software and the reliability and credibility of Level 4 and 5 assessments. Additional topics were solicited from the participants on CMM integration, measurement, technology, human issues, and quality assurance. This report contains brief summaries of the high maturity organizations participating in the workshop and the various working group reports.

http://www.sei.cmu.edu/publications/documents/00.reports/00sr003.html

CMU/SEI-2000-SR-002
ADA377375

1999 Survey of High Maturity Organizations, The

Paulk, M.; Goldenson, D.; White, D.

Over the last few years the Software Engineering Institute has investigated the high maturity practices of Maturity Level 4 and 5 software organizations via assessments, site visits, workshops, and surveys. This report summarizes the observations from the 1999 survey of high maturity organizations. Areas covered in the survey include management, engineering, tools and technology, quantitative analysis, and people issues. A specific area of interest is statistical process control, which is addressed in some detail in this report, but the observations cover a variety of engineering and management practices, including issues outside the scope of the Capability Maturity Model for Software.

http://www.sei.cmu.edu/publications/documents/00.reports/00sr002.html

Security Improvement Modules

CMU/SEI-SIM-010
ADA379469

Securing Network Servers

Allen, J.; Kossakowski, K.; Ford, G.; Konda, S.; Simmel, D.

The development of computer networks has resulted in an important class of computers: network servers. The primary purpose of these machines is to provide services, including both computational and data services, to other computers on the network.

Because of their service role, it is common for servers to store many of an organizations most valuable and confidential information resources. They also are often deployed to provide a centralized capability for an entire organization, such as communication (electronic mail) or user authentication. Security breaches on a network server can result in the disclosure of critical information or the loss of a capability that can affect the entire organization. Therefore, securing network servers should be a significant part of your network and information security strategy.

Many security problems can be avoided if servers and networks are appropriately configured. Default hardware and software configurations are typically set by vendors to emphasize features and functions more than security. Since vendors are not aware of your security needs, you must configure new servers to reflect your security requirements and reconfigure them as your requirements change.

The practices recommended here are designed to help you configure and deploy network servers that satisfy your organizations security requirements. The practices may also be useful in examining the configuration of previously deployed servers.

http://www.cert.org/security-improvement/#modules

CMU/SEI-SIM-011
ADA379920

Securing Public Web Servers

Kossakowski, K.; Allen, J.

The World Wide Web is one of the most important ways for your organization to publish information, interact with Internet users, and establish an e-commerce business presence. However, if you are not rigorous in securely configuring and operating a public Web site, you leave yourself and your organization vulnerable to a variety of security problems. You could find yourself in an embarrassing situation because malicious intruders have changed the content of your Web pages.

Compromised Web sites have served as the entry point for intrusions into an organization's internal networks for the purpose of accessing confidential information. Your organization can face business losses or legal action if an intruder successfully violates the confidentiality of customer data. Denial-of-service attacks can make it difficult, if not impossible, for users to access your Web site. This is especially critical if you are using your site to conduct business.

The practices recommended here are designed to help you mitigate the risks associated with these and several other known security problems. They build upon and assume the implementation of all practices described in the security module Securing Network Servers [Allen 00]. You need to ensure that you first configure a secure general purpose server before tailoring its configuration to operate as a public Web server.

http://www.cert.org/security-improvement/#modules

1999 Reports

Technical Reports

CMU/SEI-99-TR-032
ADA370372

Guidelines for Software Engineering Education Version 1.0

Bagert, D.; Hilburn, T.; Hislop, G.; Lutz, M.; McCracken, M.; Mengel, S.

The two central parts of the Guidelines are the description of a software engineering body of knowledge and a curriculum model. The body of knowledge presents a high-level organization and description of software engineering that supports effective curriculum design; and the curriculum model consists of a design architecture, a set of design concepts, and curriculum content guidance. We believe that this material and other guidance offered will provide assistance to faculty in the design and development of quality programs in software engineering and related curricula.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr032/99tr032abstract.html

CMU/SEI-99-TR-029
ADA001008

SRE Method Description (Version 2.0) & SRE Team Members Notebook (Version 2.0)

Williams, R.; Pandelios, G.; Behrens, S.

The Software Risk Evaluation (SRE) is a process for identifying, analyzing, and developing mitigation strategies for risks in a software-intensive system while it is in development. The SRE process has been in evolutionary development at the SEI since 1992 and has been used on over 50 Department of Defense (DoD) and civil (federal and state) contractors and program offices. Version 1.0 of the SRE Method Description was published in December, 1994.

The SRE Method Description provides

- a description of the SRE method's principles, including helpful concepts and applications
- additional insight into the SRE process so that an organization can responsibly customize the process for its own needs
- specific “key results” listings for each process step that can be used to assess quality of execution

The description should allow members of an organization’s process improvement staff to perform an initial SRE competently without outside help, and then continuously improve their process over time.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr029/99tr029abstract.html

CMU/SEI-99-TR-028
ADA375846

State of the Practice of Intrusion Detection Technologies

Allen, J.; Christie, A.; Fithen, W.; McHugh, J.; Pickel, J.; Stoner, E.

Attacks on the nation’s computer infrastructures are a serious problem. Over the past 12 years, the growing number of computer security incidents on the Internet has reflected the growth of the Internet itself. Because most deployed computer systems are vulnerable to attack, intrusion detection (ID) is a rapidly developing field. Intrusion detection is an important technology business sector as well as an active area of research. Vendors make many claims for their products in the commercial marketplace so separating hype from reality can be a major challenge. A goal of this report is to provide an unbiased assessment of publicly available ID technology. We hope this will help those who purchase and use ID technology to gain a realistic understanding of its capabilities and limitations. The report

raises issues that we believe are important for ID system (IDS) developers to address as they formulate product strategies. The report also points out relevant issues for the research community as they formulate research directions and allocate funds.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html

CMU/SEI-99-TR-027
ADA371804

Software Process Improvement Works! (Advanced Information Services Inc.)

Ferguson, P.; Leman, G.; Perini, P.; Renner, S.; Seshagiri, G.

The Advanced Information Services Inc. (AIS) Development Group was motivated by business needs to begin its Continuous Process Improvement (CPI) initiative in 1992. We chose the Software Engineering Institute (SEI) Capability Maturity Model® (CMM®) as the process maturity framework [Paulk 93] and the Institute of Electrical and Electronics Engineers (IEEE) standards as the guidelines for software engineering.

We can relate the changes in AIS process capability to three different eras. The first was the time before 1992, during which AIS did not use a model for software process improvement. During the second era, 1992 to 1995, the CMM was used. Data indicate that project schedule and effort predictability improved with the introduction of the CMM framework. The third era began in 1995 when AIS piloted the Personal Software Process^{SM} (PSP^{SM}). With the adoption of the PSP, data reflect an increase in product quality and improvements in productivity levels.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr027/99tr027abstract.html

CMU/SEI-99-TR-025
ADA 370593

Architectural Evaluation of Collaborative Agent-Based Systems

Woods, S.; Barbacci, M.

The Architecture Tradeoff Analysis Method^{SM} (ATAM^{SM}) is an architecture evaluation technique currently evolving at the Software Engineering Institute (SEI). ATAM has been applied to a number of command and control, real-time, and information systems. As collaborative, autonomous agents become a significant software technology, the demand for evaluating the quality attributes of the architectures of agent-based systems will increase. Very broadly, agents may be thought of as software entities that have the ability to undertake action autonomously in their particular embedded environment, according to a typically general set of requests or desired goals, and that are able to communicate with other agents as determined by their own initiative. Given an agent-system architecture, we need scenarios that could be applicable for conducting ATAM evaluations on instances of that agent architecture. This report identifies a few features in agent-based systems that could be used to classify agent-system architectures and to guide the generation of scenarios applicable to these architectures.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr025/99tr025abstract.html

CMU/SEI-99-TR-024
ADA373154

Builder's Guide for WaterBeans Components

Plakosh, D.; Smith, D.; Wallnau, K.

WaterBeans is a proof-of-feasibility system for building software applications through a process of assembling (composing) prefabricated software components. WaterBeans was originally developed as a proof of feasibility that software component technology could be used to develop software applications in the domain of water-quality modeling. (In particular, WaterBeans supports modeling and simulating the hydrology of urban storm water sewage and runoff.) WaterBeans includes a component model for component developers, a visual composition environment for importing and assembling components into applications, and several families of components. One family of components supports modeling and simulating urban sewage systems. Another family of components was developed to prove the generality of WaterBeans; this family of components allows visualization and manipulation of digital waveforms. This report documents the programming interface for component developers. It also provides a brief description of the composition environment.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr024/99tr024abstract.html

CMU/SEI-99-TR-023
ADA373286

Case Study on Analytical Analysis of the Inverted Pendulum Real-Time Control System, A

Seto, D.; Sha, L.

An inverted pendulum has been used as the controlled device in a prototype real-time control system employing the Simplex^{TM} architecture. In this report, we address the control issues of such a system in an analytic way. In particular, an analytic model of the system is derived; control algorithms are designed for the baseline control, experimental control and safety control based on the concept of analytic redundancy; the safety region is obtained as the stability region of the system under the safety control; and the control switching logic is established to provide fault tolerant functionality. Finally, the results obtained and the lessons learned are summarized, and future work is discussed.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr023/99tr023abstract.html

CMU/SEI-99-TR-022
ADA371802

Attribute-Based Architectural Styles

Klein, M.; Kazman, R.

An architectural style is a description of component types and their topology. It also includes a description of the pattern of data and control interaction among the components and an informal description of the benefits and drawbacks of using that style. Architectural styles are important engineering artifacts because they define classes of designs along with their associated known properties. They offer experience-based evidence of how each class has been used historically, along with qualitative reasoning to explain why each class has its specific properties.

Attribute-Based Architectural Styles (ABASs) build on architectural styles to provide a foundation for more precise reasoning about architectural design by explicitly associating a reasoning framework (whether qualitative or quantitative) with an architectural style. These reasoning frameworks are based on quality attribute-specific models, which exist in the various quality attribute communities (such as the performance and reliability communities).

Architectural styles, and hence ABASs, are powerful because they provide a designer with the concentrated wisdom of many preceding designers faced with similar problems. In this report we exemplify the use of ABASs in both design and analysis. We argue that ABASs provide the groundwork to create an engineering discipline of architectural design—to make design a predictable process rather than an ad hoc one.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr022/99tr022abstract.html

CMU/SEI-99-TR-021
ADA367575

Information Assurance Curriculum and Certification: State of the Practice

Laswell, B.; Simmel, D.; Behrens, S.

The purpose of this document is to describe the state of the practice in information assurance and security curriculum and certification. The scope is not exhaustive, but rather illustrative of the types of activity occurring today within various organizations, including government, universities and research centers, professional societies, and the business community. Although individual courses are available, there apparently is no systematic agreement on the knowledge, skills, and abilities required to formulate a curriculum for information security professionals that enjoys broad-based support across organizations. As a result of Presidential Decision Directive 63 and the charge to protect the nation's critical infrastructures, the pressure is increasing to provide some minimum level of competence for system and network administrators working in the field of information assurance. Presently, several professional organizations offer certified professional designations.

What is needed is a comprehensive framework for curriculum and certification in information assurance and security. Currently the thrust for training focuses primarily on the technologies of information infrastructures. However, long-term solutions for the protection of critical information assets will require a more comprehensive approach in which senior executives and managers, as well as technical staff, develop strong and diverse skills that allow them to advance an organization's mission in a dynamic and increasingly hostile networked environment.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr021/99tr021abstract.html

CMU/SEI-99-TR-020
ADA379857

Case Study: Development of a Baseline Controller for Automatic Landing of an F-16 Aircraft Using Linear Matrix Inequalities (LMIs)

Seto, D.; Ferriera, E.; Marz, T.

In this report, we present preliminary results on the design of the baseline controller for an F-16 aircraft automatic landing system using linear matrix inequalities (LMI)-based approaches. We start with a general study of aircraft control and dynamics to gain knowledge of the structure of an aircraft dynamic model and its inner loop control system. We then identify a linear model along the glide path for the inner loop control system in the simulator. With this linear model, the control objective is to solve a stabilization problem—stabilizing the aircraft along the glide path using linear state feedback controls. Expressing the stability criterion and the constraints in LMIs, we cast the stabilization problem as an optimization problem. Using the SDPSOL software package developed by Wu and Boyd, we solve this optimization problem for the control gain and stability region, which completes the controller design [Wu 96].

http://www.sei.cmu.edu/publications/documents/99.reports/99tr020/99tr020abstract.html

CMU/SEI-99-TR-018
ADA367624

Engineering Method for Safety Region Development, An

Seto, D.; Sha, L.

In this report, we study tolerance of semantic faults, one of the crucial issues in the Simplex^{TM} architecture. In particular, we examine semantic faults that cause the controlled device to be unsafe (i.e., unable to carry out its normal operation) and eventually cause the device to become damaged. We also consider fault detection as a safety check. For the class of control systems operating around an equilibrium, the objective of maintaining the safety of the controlled device is formulated as a stabilization problem, and the safety of the controlled device is tested against the stability region of the device under the safety control. To establish the stability region, we apply the Lyapunov stability theorem and linear matrix inequality (LMI) methodologies. It is shown that the stability region for a given safety controller as well as a safety control law can be systematically derived using LMI-based approaches. We conclude the report with a summary of the procedure for deriving the safety check and safety controller for a given application.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr018/99tr018abstract.html

CMU/SEI-99-TR-017
ADA367718

Operationally Critical Threat, Asset, and Vulnerability Evaluation^{SM} (OCTAVE^{SM}) Framework, Version 1.0

Alberts, C.; Behrens, S.; Pethia, R.; Wilson, W.

The Operationally Critical Threat, Asset, and Vulnerability Evaluation^{SM} (OCTAVE^{SM}) is a framework for identifying and managing information security risks. It defines a comprehensive evaluation method that allows an organization to identify the information assets that are important to the mission of the organization, the threats to those assets, and the vulnerabilities that may expose those assets to the threats. By putting together the information assets, threats, and vulnerabilities, the organization can begin to understand what information is at risk. With this understanding, the organization can design and implement a protection strategy to reduce the overall risk exposure of its information assets.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr017/99tr017abstract.html

CMU/SEI-99-TR-016

Simplex in a Hostile Communications Environment: The Coordinated Prototype

Sha, L.; Seto, D.Altman, N.; Weinstock, C.; Sha, L.; Seto, D.

Simplex is an engineering framework embodying fault tolerance and dynamic upgrade in a highly maintainable system. This report describes an approach to using Simplex to construct a COTS-based computer system capable of coordinated real-time motion control in a hostile communications environment. It also discusses how selected portions of the Coordinated Prototype tolerate software faults and allow new software to be deployed and tested during system operation.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr016/99tr016abstract.html

CMU/SEI-99-TR-015
ADA375845

Second DoD Product Line Practice Workshop Report

Bergey, J.; Campbell, G.; Clements, P.; Cohen, S.; Jones, L.; Krut, R.; Northrop, L.; Smith, D.

The Software Engineering Institute (SEI) held the Second Department of Defense (DoD) Product Line Practice Workshop in March 1999. The workshop was a hands-on meeting to identify industry-wide best practices in software product lines; to share DoD product line practices, experience, and issues; and to discuss ways in which the current gap between commercial best practice and DoD practice can be bridged. This report synthesizes the workshop presentations and discussions.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr015/99tr015abstract.html

CMU/SEI-99-TR-014

Architecture Tradeoff Analyses of C4ISR Products

Barbacci, M.; Wood, W.

Early evolution of the architecture of a system or a product line of systems is a low-cost risk reduction method for determining whether the system(s) will achieve its business and quality goals. The Architecture Tradeoff Analysis Method (ATAM) is an architecture evaluation technique currently evolving at the Software Engineering Institute (SEI). The input to the ATAM consists of a system or product line architecture and the perspectives of stakeholders involved with that system or product line. The output of the ATAM is (1) a collection of scenarios that help specify the context of the system's or product line's use and the product line's evolution, (2) improved architectural documentation (usually), and (3) analysis results (in particular, a set of issues to consider, risks, and potential sensitivity and tradeoff points within the architecture). Currently, there are no generally accepted industry-wide standards for describing a system architecture, and ATAM evaluations are often tailored to the available documentation.

The Architectures Directorate of the C4I Integration Support Activity (CISA), Office of the Assistant Secretary of Defense for Command, Control, Communications, and Intelligence (OASD[C3I]) has defined a framework for architecture development, presentation, and integration to be used across the military services and defense agencies. This framework for Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) is becoming the required method for describing information systems with the Department of Defense (DoD) and other U.S. Government agencies. This report describes how various C4ISR products can be used in the context of an ATAM evaluation and their relative value for generating quality attribute-specific scenarios required for an ATAM evaluation.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr014/99tr014abstract.html

CMU/SEI-99-TR-013
ADA373330

Construction and Deployment Scripts for COTS-Based, Open Source Systems

Hansen, W.

Construction/deployment scripts direct the compilation of sources to executables and the installation of those executables. This report details the construction/deployment scripts developed at the Software Engineering Institute for the GEE project. GEE, a Generic Enterprise Ensemble, is a prototypical three-tier information system incorporating a number of commercial off-the-shelf (COTS) products. The scripts for GEE were challenging because we wanted a self-contained package of scripts and source files from which the system could be built and deployed either by us at our site or by customers at their sites. The COTS products we used—Java, an Oracle database, the Visibroker implementation of CORBA (the Common Object Request Broker Architecture), and the Netscape Web browser and server—added challenges in installation, version change, process initiation, and communication rendezvous. This report describes the challenges and how our solutions exploited the principles of "Repeat not" and "Delay binding." Lessons learned are reported elsewhere.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr013/99tr013abstract.html

CMU/SEI-99-TR-012
ADA366095

Why Do Organizations Have Assessments? Do They Pay Off?

Dunaway, D.; Berggren, R.; Rochettes, G.; Iredale, P.; Lavi, I.; Taylor, G.

The authors of this report appeared on a panel at SEPG '99 (1999 Software Engineering Process Group Conference) on March 10, 1999, in Atlanta, Georgia. Each panelist is one of the "most active lead assessors" in the Software Engineering Institute (SEI) Appraiser Program and has conducted and reported on over 5 Capability Maturity Model® (CMM®) -Based Appraisals for Internal Process Improvement (CBA IPIs) over the past 18 months. In this report, the panelists document their experiences regarding why an organization chooses to have a CBA IPI and what the organization gains from having conducted an assessment. Each author discusses the experiences related to CBA IPIs that he or she has led with a focus on the organizational perspective before, during, and after the assessment. Since a CBA IPI is a major organizational intervention, it is important to know why an organization has an assessment, if it has a positive impact on an organization, and how an organization would characterize the benefits that the assessment provides. What needs to be done during the planning, how the assessment is conducted, and what follows the assessment are all important factors that will affect the benefits that the organization experiences from the assessment.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr012/99tr012abstract.html

CMU/SEI-99-TR-010
ADA362 725

Why Reengineering Projects Fail

Bergey, J.; Smith, D.; Tilley, S.; Weiderman, N.; Woods, S.

The purpose of this report is to highlight some of the most important reasons for failures in reengineering efforts despite the best of intentions. We support our observations with examples from a variety of experiences over many years. Readers may recognize some of the situations presented here and be tempted to conclude that the examples are taken from their own organizations, but similar missteps occur quite frequently.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr010/99tr010abstract.html

CMU/SEI-99-TR-009
ADA375848

Rollout and Installation of Risk Management at the IMINT Directorate, National Reconnaissance Office

Loveland Link, J.; Barbour, R.; Krum, A.; Neitzel, A.

The NRO Risk Management pilot project, and subsequent rollout and installation, were launched in the Imagery Development Program (IDP) at the Imagery Intelligence (IMINT) Directorate. This was preceded by a Software Acquisition Capability Maturity Model® (SA-CMM®) assessment to determine strengths and gaps in IMINT's capability as an acquisition organization. From the potential SA-CMM improvement areas, IMINT leaders determined that the optimum first initiative would be Acquisition Risk Management. To launch the Risk Management initiative, IMINT leaders identified the Command and Control Division (CCD) in IDP as the pilot initiative. They further decided to conduct Software Risk Evaluations (SREs) with both the government organization and the principal contractor for CCD. The CCD division proceeded to install a dynamic, interactive Risk Management process through-out its program, with a Team Risk Management approach. This approach was leveraged by monthly CCD Team Risk Reviews (TRRs). The TRRs served as regular forums for government and contractors to identify and mitigate joint risks.

The IDP director, observing the success of the CCD pilot, called for consistent Risk Management training across the divisions. Concurrently, the IDP director also launched the EIS System Risk Management Team (ESRT) with a mission to identify and address system-level risks. The divisional Risk Management processes (including contractor Risk Management) were combined with system-level Risk Management at the ESRT to develop a comprehensive, consistent Risk Management process that became routinely operational.

The central purpose of this report is to provide a knowledge asset repository for the National Reconnaissance Office that can be leveraged in support of further Risk Management efforts.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr009/99tr009abstract.html

CMU/SEI-99-TR-008
ADA367714

Guidelines for Developing a Product Line Concept of Operations

Cohen, S.

This report provides guidelines for an organization that is developing a Concept of Operations (CONOPS) document. A CONOPS document defines the organization's product line approach. The CONOPS document and the decisions made in its preparation will guide the organization as it plans and executes the process of fielding a product line, from product line scoping, through architecture, component development, and product development.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr008/99tr008abstract.html

CMU/SEI-99-TR-007
ADA366100

Architecture-Based Development

Bass, L.; Kazman, R.

This report presents a description of architecture-centric system development. In an architecture-centric process, a set of architecture requirements is developed in addition to functional requirements. This report describes the source of these architecture requirements and how they are elaborated into a design. In addition to design, the documentation, evaluation, realization, and maintenance of an architecture are also described.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr007/99tr007abstract.html

CMU/SEI-99-TR-006
ADA373332

Lessons Learned Collaborating on a Process for SPI at Xerox

Fowler, P.; Middlecoat, B.; Yo, S.

During 1995-1998, Xerox Corporation's West Coast Production Systems Group (PSG West) worked with the Software Engineering Institute (SEI) to apply the prototype Process Change Model (PCM) to aid in their efforts to reach Level 2 of the Software Capability Maturity (and to develop the generic processes required for Level 3. The Process Change Model, along with a companion guidebook, was designed to provide the basis for a systematic approach to technology-specific change based in part on "whole product" principles, with a focus on one key process area (KPA) at a time. This report describes a collaborative effort to develop a more systematic and detailed approach to software process improvement (SPI) through use and evaluation of prototype versions of the PCM and guidebook. In particular, the work of the PSG West software engineering process group (SEPG) to apply the PCM and guidebook in working with improvement action teams focused on the KPAs of the Software CMM is described. Lessons learned about the "live" evaluation and maturation of a new process and guidebook such as this are presented. These lessons should be of interest to those engaged in work on technology maturation and the adoption of technological or process innovations as well as to those engaged in SPI and process development.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr006/99tr006abstract.html

CMU/SEI-99-TR-005
ADA366089

Introduction to Software Engineering Practices Using Model-Based Verification, An

Gluch, D.; Brockway, J.

This is an introductory report on the use of model-based verification techniques within software development and upgrade practices. It presents the specific activities and responsibilities that are required of engineers who use the model-based verification paradigm and describes proposed approaches for integrating model-based verification into an organization's software engineering practices. The approaches outlined in this report are preliminary concepts for the integration of model building and analysis techniques into software engineering review and inspection practices. These techniques are presented as both practices within peer review processes and as autonomous engineering investigations. The objective of this report is to provide a starting point for the use of model-based verification techniques and a framework for their evaluation in real-world applications. It is expected that the results of pilot studies that employ the preliminary approaches described here will form the basis for improving the practices themselves and software verification generally.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr005/99tr005abstract.html

CMU/SEI-99-TR-004

Software Engineering Body of Knowledge Version 1.0, A

Hilburn, T.; Hirmanpour, I.; Khajenoori, S.; Turner, R.; Qasem, A.

Software engineering, both as a discipline and as a profession, is at a pivotal point in its evolution. Although software has become critical in the development of most new human-created systems, the concepts, principles, and methods for engineering software are still neither well defined nor uniformly agreed upon. The lack of consensus regarding software engineering practice and the requisite competencies creates confusion and has serious consequences for the evaluation, acquisition, and application of software engineering knowledge. This report presents an effort to organize and catalogue a body of knowledge for software engineering and to provide a systematic, concise, and complete description of the software engineering discipline. This body of knowledge can assist organizations in defining and improving the software engineering competencies of their workforces; it can help educational institutions in defining software engineering curricula; it can provide a basis for classifying academic and industrial research and development efforts; and it can improve the understanding and practice of software engineering.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr004/99tr004abstract.html

CMU/SEI-99-TR-003
ADA361391

Third Product Line Practice Workshop Report

Bass, L.; Campbell, G.; Clements, P.; Northrop, L.; Smith, D.

The Third Software Engineering Institute Product Line Practice Workshop was a hands-on meeting held in December 1998 to share industry practices in software product lines, to explore the technical and non-technical issues involved, and to evolve the SEI Product Line Practice Framework. This report synthesizes the workshop presentations and discussions, which described product line practices and analyzed issues in the areas of software engineering, technical management, and organizational management.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr003/99tr003abstract.html

CMU/SEI-99-TR-002
ADA362667

Software Acquisition Capability Maturity Model (SA-CMM), Version 1.02

Cooper, J.; Fisher, M.; Sherer, S.W. (eds.)

Government and industry have the need to improve the maturity of their internal software acquisition processes. In order for organizations to make improvements, they must know the ultimate goal and what is required to achieve that goal. Additionally, progress toward achieving the goal must be measurable. A capability maturity model provides the framework needed to facilitate the desired improvement. The Software Acquisition Capability Maturity Model (SA-CMM) has been developed to provide such a framework.

This new version incorporates change requests that have been received, as well as the results of lessons learned from conducting appraisals and from the use of Version 1.01.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr002/99tr002abstract.html

CMU/SEI-99-TR-001

Building Blocks for Achieving Quality of Service with Commercial Off-the-Shelf (COTS) Middleware

Polze, A.

To date, most of the fault-tolerant, real-time systems have been implemented in embedded settings, and there is an urgent need to open up this type of computing technology to a larger number of people who use heterogeneous distributed computing environments. Today's transportation, manufacturing, and communication systems require the integration of multiple embedded real-time control systems with standard distributed computing environments in a predictable fashion. Humboldt University has developed the concept of *composite objects* as a filtering bridge between standard middleware platforms and software frameworks providing services with certain quality-of-service (QoS) guarantees. Current research focuses on the Common Object Request Broker Architecture (CORBA) middleware platform; however, composite objects are also applicable to platforms like the Distributed Component Model (DCOM) and distributed computing environments (DCEs). Key concepts in Humboldt's approach are analytic redundancy, noninterference, interoperability, and ADAptive abstraction. These concepts originated in SEI work on

the Simplex architecture and have been reapplied to extend the reach of commercial off-the-shelf (COTS) software technologies into demanding application settings (such as those found in military and industrial applications). Here, we discuss building blocks and techniques for fault-tolerant, real-time applications based on CORBA.

http://www.sei.cmu.edu/publications/documents/99.reports/99tr001/99tr001abstract.html

Technical Notes

CMU/SEI-99-TN-015
ADA379746

Lessons Learned Applying Commercial Off-the-Shelf Products Manufacturing Resource Planning II Program

Brownsword, L.; Place, P.

While the lure of easy system construction from pre-existing building blocks that snap into place is appealing, current reality reveals a less than ideal picture, particularly for commercial off-the-shelf (COTS) software components. Examining the similarities and differences of organizations that have applied COTS and the successes and failures of those organizations has enabled the COTS-Based Systems (CBS) Initiative at the Software Engineering Institute (SEI) to identify a number of significant capabilities that an organization must have to succeed with a COTS-based approach. This case study of the Manufacturing Resource Planning II program is part of a series of case studies that seek to identify important acquisition, business, and engineering issues surrounding the use of COTS-based systems and thus derive available solutions, where possible.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn015/99tn015abstract.html

CMU/SEI-99-TN-014
ADA370 600

Options Analysis for Reengineering (OAR): Issues and Conceptual Approach

Bergey, J.; Smith, D.; Weiderman, N.; Woods, S.

Organizations that own or use software assets require a structured and validated approach for making decisions on how to update, migrate, or reengineer their legacy assets. A model has recently been developed to understand technical transformations at different levels of abstraction. However, this model, which focuses on technical issues, is not yet accessible for decision-makers. This report outlines the foundation of a structured and coherent method, based on the "horseshoe" model, that will help practitioners make appropriate reengineering choices.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn014/99tn014abstract.html

CMU/SEI-99-TN-013
ADA370621

DoD Legacy System Migration Guidelines

Bergey, J.; Smith, D.; Weiderman, N.

DoD systems contain a substantial amount of legacy software that often needs to be evolved or migrated to new hardware and software platforms. This legacy software is important not only because of the large-scale investment it represents, but also because it defines the baseline for describing future desired capabilities and evolutionary growth. The purpose of this report is to provide a set of DoD legacy system migration guidelines. These guidelines are built on lessons learned from actual legacy system migration efforts and they synthesize previous work on the disciplined evolution of legacy systems.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn013/99tn013abstract.html

CMU/SEI-99-TN-012
ADA377450

Software Architecture with ATAMSM in the DoD System Acquisition Context

Bergey, J.; Fisher, M.; Jones, L.; Kazman, R.

Many modern defense systems rely heavily on software to achieve system functionality. Because software architecture is a major determinant of software quality, it follows that software architecture is critical to the quality of a software-intensive system. For a Department of Defense (DoD) acquisition organization, the ability to evaluate software architectures can have a favorable impact on the delivered system. This technical note explains the basics of software architecture and software architecture evaluation in a system-acquisition context. It also sets the context for applying software architecture evaluation based on the Architecture Tradeoff Analysis Method^{SM} (ATAM^{SM}) in the DoD acquisition environment. Future versions of this technical note will expand upon this conceptual approach and provide additional details drawn from real experiences.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn012/99tn012abstract.html

CMU/SEI-99-TN-011
ADA373184

Product Line Acquisition in the DoD: The Promise, The Challenges

Jones, L.

Industrial use of software product line technology has resulted in some impressive savings, while also improving product quality and delivery time. Although there has been some successful use of this technology within the Department of Defense (DoD), there are special challenges. This paper presents the basics of product line practices and reports the results of two DoD product line workshops in which important issues and successful practices were shared.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn011/99tn011abstract.html

CMU/SEI-99-TN-010

Into the Black Box: A Case Study in Obtaining Visibility into Commercial Software

Plakosh, D.; Hissam, S.; Wallnau, K.

We were recently involved with a project that faced an interesting and not uncommon dilemma. The project needed to programmatically extract private keys and digital certificates from the Netscape Communicator v4.5 database. Netscape documentation was inadequate for us to figure out how to do this. As it turns out, this inadequacy was intentional-Netscape was concerned that releasing this information might possibly violate export control laws concerning encryption technology. Since our interest was in building a system and not exporting cryptographic technology, we decided to further investigate how to achieve our objectives even without support from Netscape. We restricted ourselves to the use of Netscape-provided code and documentation, and to information available on the Web. Our objective was to build our system, and to provide feedback to Netscape on how to engineer their product to provide the capability that we (and others) need, while not making the product vulnerable or expose the vendor to violations of export control laws. This paper describes our experiences peering “into the black box.”

http://www.sei.cmu.edu/publications/documents/99.reports/99tn010/99tn010abstract.html

CMU/SEI-99-TN-006
ADA366097

Custom vs. Off-the-Shelf Architecture

Seacord, R.; Wallnau, K.; Robert, J.; Comella-Dorda, S.; Hissam, S.

Members of the COTS-Based Systems Initiative at the Software Engineering Institute have developed the Generic Enterprise Ensemble (GEE), a generic approach to building distributed, transaction-based, secure enterprise information systems (EIS). GEE is a tool to help in the selection of technologies and architectural choices when building Enterprise Information Systems. Enterprise JavaBeansTM (EJB) is a specification from Sun Microsystems for an application server based on Java technology. In this paper, a comparison is made between GEE-based solutions and off-the-shelf solutions based on the EJB specification.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn006/99tn006abstract.html

CMU/SEI-99-TN-005

Theory and Practice of Enterprise JavaBean Portability

Comella-Dorda, S.; Robert, J.; Seacord, R.

The modern enterprise information system (EIS) requires the integration of numerous technologies such as distribution, transactions, data management, security, and naming. Off-the-shelf architectures such as Enterprise JavaBeans (EJB) provide a pre-integrated solution that supports the quick development and deployment of information systems. Unfortunately, the EJB specification is extremely porous, leading to portability problems. In addition, the line between vendor extensions and EJB standard functionality is blurred, making it difficult for bean providers to know what functionality can be depended upon across server implementations. This paper presents sources of portability problems in EJB and illustrates them with some real examples. We also present our opinion about the direction the EJB specification should take to enable effective reuse of Enterprise Beans between servers.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn005/99tn005abstract.html

CMU/SEI-99-TN-004

DoD Acquisition Environment and Software Product Lines, The

Bergey, J.; Fisher, M.; Jones, L.

Industrial experience clearly demonstrates that a product line approach for software-intensive systems can save money and result in faster time to field higher quality systems. Many within the DoD recognize the benefits of product lines, but also recognize that there are significant challenges to adopting this approach. Many of these challenges stem from the fact that the DoD is in the business of acquiring systems rather than developing them.

A key question is how can a software product line approach best be accommodated within the current DoD acquisition environment? In order to answer this question, this technical note examines three key DoD acquisition policies and regulations and their implications for launching a product line approach. This includes examining the DoD acquisition management process and DoD guidance on acquisition strategies that set the context for software product line acquisition planning. Sources of confusing guidance on developing acquisition strategies are examined and terms are defined to clarify what is meant by a product line acquisition strategy. The need for strategic acquisition planning in launching a product line is discussed and insight is provided on how it differs from traditional acquisition planning.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn004/99tn004abstract.html

CMU/SEI-TN-003
ADA366088

COTS in the Real World: A Case Study in Risk Discovery and Repair

Hissam, S.; Plakosh, D.

Like many organizations in both the public and private sectors, the U.S. Department of Defense (DoD) is committed to a policy of using commercial off-the-shelf (COTS) components in new systems, particularly information systems. However, the DoD also has a long-standing set of security needs for its systems, and the pressure to adopt COTS components can come into conflict with those security constraints. The major elements of this conflict are the DoD's overall approach to system security on one hand and the economic forces that drive the component industry on the other. As DoD managers and system integrators look to the COTS marketplace for components to satisfy more security requirements, this conflict becomes more prominent. In this report, we describe an actual product evaluation where just such a conflict occurred, examine why that conflict exists, and outline the corrective steps that were taken.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn003/99tn003abstract.html

CMU/SEI-99-TN-002

Securing Internet Sessions with Sorbet

Long, F.; Hissam, S.; Seacord, R.; Robert, J.; Robert, J.

More and more organizations are using intranets, and even the Internet, as the communications media for important data. However, such communications media are inherently insecure and subject to hijacking. To secure these connections, mechanisms must be built on top of the underlying communications facilities. In this paper, we discuss one such security mechanism and describe an implementation using common object request broker architecture (CORBA) -based interceptors.

http://www.sei.cmu.edu/publications/documents/99.reports/99tn002/99tn002abstract.html

Special Reports

Quotations from Chairman David (A Little Red Book of Truths to Enlighten and Guide on the Long March Toward the COTS Revolution)

Carney, D.

Quotations from Chairman David is a brief and humorous examination of some issues related to commercial off-the-shelf (COTS) products in DoD and government systems. The author, David Carney, has borrowed from a most unusual historical source—a certain "Little Red Book" popular in the 1960s—and put together some useful observations on some of the facts and fictions that underlie the current interest in COTS-based systems.

http://www.sei.cmu.edu/publications/documents/99.reports/lrb/lrb-abstract.html

CMU/SEI-99-SR-006
ADA363791

Analysis of Courses in Information Management and Network System Security and Survivability

Capell, P.

This report provides an overview of instructional systems design and its implications for analyzing curricula in the fields of information management and networked systems longevity. Measurable benchmarks for assessment of training and educational resources are offered in order to fully illustrate how to perform instructional gap analysis. This report also addresses issues of instructional approaches and metrics, performance objectives, educational measurement, and mission vs. learning objectives, and includes a selected listing of related coursework in the appendices.

http://www.sei.cmu.edu/publications/documents/99.reports/99sr006/99sr006abstract.html

CMU/SEI-99-SR-001
ADA360577

Directory of Industry and University Collaborations with a Focus on Software Engineering Education and Training, Version 7

Beckman, K.

This directory describes 23 formal collaborative efforts to promote software engineering education and training activities among industry organizations (including government) and universities in the United States, Can ADA, and Australia. These collaborations vary in their organizational structure and types of services offered. All attempt to bridge the gap between industry needs and academic software engineering education and training offerings. Readers can use this directory to find collaborations that match their software engineering and training needs and that are located in their geographic area.

http://www.sei.cmu.edu/publications/documents/99.reports/99sr001/99sr001abstract.html

Handbooks

CMU/SEI-99-HB-001
ADA370385

Software Acquisition Risk Management Key Process Area (KPA)—A Guidebook Version 1.02

Gallagher, B.

In this guidebook, we provide sponsors of acquisition improvement programs and their immediate staff with guidelines on how to implement a software acquisition risk management program satisfying the goals of the Acquisition Risk Management (ARM) Key Process Area (KPA) of the Software Acquisition Capability Maturity Model^{SM} (SA-CMM^{SM}). Brief overviews of software acquisition and the SA-CMM are included. This version is an editorial update to align with Version 1.02 of the SA-CMM.

http://www.sei.cmu.edu/publications/documents/99.reports/99hb001/99hb001abstract.html

Security Improvement Modules

CMU/SEI-SIM-008
ADA367717

Deploying Firewalls

Fithen, W.; Allen, J.; Stoner, E.

A firewall is a combination of hardware and software used to implement a security policy governing the network traffic between two or more networks, some of which may be under your administrative control (e.g., your organizations networks) and some of which may be out of your control (e.g., the Internet). A network firewall commonly serves as a primary line of defense against external threats to your organization's computer systems, networks, and critical information. Firewalls can also be used to partition your organizations internal networks, reducing your risk from insider attacks.

Firewall technologies have entered into the mainstream. The [Power 99] indicates that 91 percent of the organizations surveyed already deploy firewalls. Articles and other references covering evaluation, selection, and configuration of firewall technologies are now common in the popular press (see References at the end of this section). However, there has been little published about designing, installing, deploying, operating, and maintaining firewalls. The practices in this module will address designing, installing, and deploying firewalls.

The term firewall is taken from the structural analog whose purpose is to slow the spread of fire in a building. In the computer literature, popular press, and vendor marketing materials, the term is used in many ways. Some people use it to identify a specific hardware component or software package, while others consider the entire collection of systems and software deployed between two networks to be parts of a firewall.

Throughout these practices, we will generally use the term firewall as an adjective modifying a noun (such as system, hardware, software, product) to make the reference clear. When we use the term firewall as a noun, we mean the general concept of a technological mechanism for the enforcement of a network traffic security policy. While this may seem cumbersome at times, we believe these distinctions will increase your understanding of our intent.

http://www.cert.org/security-improvement/#modules

CMU/SEI-SIM-007
ADA361387

Securing Network Servers

Ford, G.; Allen, J.; Alberts, C.; Fraser, B.; Hayes, E.; Kochmar, J.; Konda, S.; Kossakowski, K.; Simmel, D.; Vermeulen, D.

The development of computer networks has resulted in an important class of computers: network servers. The primary purpose of these machines is to provide services, including both computational and data services, to other computers on the network. Because of their service role, it is common for servers to store many of an organizations most valuable and confidential information resources. They also are often deployed to provide a centralized capability for an entire organization, such as communication (electronic mail) or user authentication. Security breaches on a network server can result in the disclosure of critical information or the loss of a capability that can affect the entire organization. Therefore, securing network servers should be a significant part of your network and information security strategy. Many security problems can be avoided if servers and networks are appropriately configured. Default hardware and software configurations, however, are set by vendors who tend to emphasize features and functions more than security. Since vendors are not aware of your security needs, you must configure new servers to reflect your security requirements and reconfigure them as your requirements change. The practices recommended here are designed to help you configure and deploy network servers that satisfy your organizations security requirements. The practices may also be useful in examining the configuration of previously deployed servers.

http://www.cert.org/security-improvement/#modules

CMU/SEI-SIM-006
ADA360500

Responding to Intrusions

Kossakowski, K.; Allen, J.; Alberts, C.; Cohen, C.; Ford, G.; Fraser, B.; Hayes, E.; Kochmar, J.; Konda, S.; Wilson, W.

These practices are intended primarily for system and network administrators, managers of information systems, and security personnel responsible for networked information resources. These practices are applicable to your organization if your networked systems infrastructure includes host systems providing services to multiple users (file servers, timesharing systems, database servers, Internet servers, etc.) local-area or wide-area networks direct connections, gateways, or modem access to and from external networks, such as the Internet We recommend that

you read all of the practices in this module before taking any action. To successfully implement the practices, it is important that you understand the overall context and relationships among them. For instance, once you read the practices in the Handle category, it is easier to understand the Practices in the Prepare category (see the Summary of recommended practices table). If you are dealing with an intrusion, you may want to skip the first two preparatory practices and move immediately to Practice 3, “Analyze all information necessary to characterize an intrusion.” Once you have completed your response and recovery process, we recommend that you review and implement the preparatory practices.

http://www.cert.org/security-improvement/#modules

CMU/SEI-SIM-004
ADA361388

Securing Desktop Workstations

Simmel, D.; Ford, G.; Allen, J.; Alberts, C.; Fraser, B.; Hayes, E.; Kochmar, J.; Konda, S.;

The development of computer networks has resulted in an important class of computers: network servers. The primary purpose of these machines is to provide services, including both computational and data services, to other computers on the network. Because of their service role, it is common for servers to store many of an organizations most valuable and confidential information resources. They also are often deployed to provide a centralized capability for an entire organization, such as communication (electronic mail) or user authentication. Security breaches on a network server can result in the disclosure of critical information or the loss of a capability that can affect the entire organization. Therefore, securing network servers should be a significant part of your network and information security strategy. Many security problems can be avoided if servers and networks are appropriately configured. Default hardware and software configurations, however, are set by vendors who tend to emphasize features and functions more than security. Since vendors are not aware of your security needs, you must configure new servers to reflect your security requirements and reconfigure them as your requirements change. The practices recommended here are designed to help you configure and deploy network servers that satisfy your organizations security requirements. The practices may also be useful in examining the configuration of previously deployed servers.

http://www.cert.org/security-improvement/#modules

1998 Reports

Technical Reports

CMU/SEI-98-TR-017
ADA358797

rlogin(1): The Untold Story

Rogers, L.W.

Coding defects account for a significant portion of the reports received by the CERT Coordination Center® (CERT/CC). Through in-depth analysis of these reports and generalizing our findings from those analyses, we have begun to create guidelines for mitigation strategies for existing defects and avoidance strategies when coding new software. In this document, we report the results of our analysis of the well-known defect in the rlogin program. We discuss the coding defect in detail, three mitigation strategies devised to remedy the defect, and two avoidance strategies offered as a guide to reducing the instances of similar coding defects in new programs. We end with three design notes aimed at eliminating these defects at the hardware and protocol design level.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr017/98tr017abstract.html

CMU/SEI-98-TR-015
ADA354691

Second Product Line Practice Workshop Report

Bass, L.; Chastek, G.; Clements, P.; Northrop, L.; Smith, D.; Withey, J.

The second Software Engineering Institute Product Line Practice Workshop was a hands-on meeting held in November 1997 to share industry practices in software product lines and to explore the technical and non-technical issues involved. This report synthesizes the workshop presentations and discussions, which identified factors involved in product line practices and analyzed issues in the areas of software engineering, technical management, and enterprise management.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr015/98tr015abstract.html

CMU/SEI-98-TR-014
ADA355070

Case Study in Survivable Network System Analysis

Ellison, R.; Linger, R.; Longstaff, T.; Mead, N.

This paper presents a method for analyzing the survivability of distributed network systems and an example of its application. Survivability is the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents. Survivability requires capabilities for intrusion resistance, recognition, and recovery. The Survivable Network Analysis (SNA) method builds on the Information Security Evaluation previously developed by permitting assessment of survivability strategies at the architecture level. Steps in the SNA method include system mission and architecture definition, essential capability definition, compromisable capability definition, and survivability analysis of architectural softspots that are both essential and compromisable. Intrusion scenarios play a key role in the method. SNA results are summarized in a Survivability Map which links recommended survivability strategies for resistance, recognition, and recovery to the system architecture and requirements. This case study summarizes the application and results of applying the SNA method to a subsystem of a large-scale, distributed healthcare system. The study recommended specific modifications to the subsystem architecture to support survivability objectives. Positive client response to study recommendations suggests that the method can provide significant added value for ensuring survivability of system operations. As a result of this case study, the SNA method, artifacts, and lessons learned will be available to apply architectural analysis for survivability to proposed and legacy DoD distributed systems.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr014/98tr014abstract.html

CMU/SEI-98-TR-013
ADA358781

Study of Practice Issues in Model-Based Verification Using the Symbolic Model Verifier (SMV), A

Srinivasan, G.R.; Gluch, D.

This report presents the results of a study on the practice issues involved in using the Symbolic Model Verifier (SMV) for model checking software systems. The case study is of a Simplex implementation: the Simplex coordinated demonstration system for reliable system upgrade. The investigation consisted of generating a system model (using both statechart and SMV notations), specifying claims (expected properties) of the system as temporal logic formulae, and checking those formulae with respect to the SMV model. The various steps involved in the modeling process are described. Examples of the claims, their results, and a description of how the SMV tool analyzed them are detailed. Key engineering decisions made during the modeling process and a work breakdown of the effort are also presented.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr013/98tr013abstract.html

CMU/SEI-98-TR-012
ADA354685

People CMM®-Based Assessment Method Description

Hefley, W.; Curtis, B.

This document provides a high-level overview of the People Capability Maturity Model℠ (CMM®)-Based Assessment Method. It introduces the People CMM as a source of guidelines for improving the capability and readiness of an organization's workforce in the context of the IDEAL℠ approach to process improvement. In order to measure the capability and maturity of an organization's workforce practices, an appraisal method has been developed for the People CMM. This document describes the requirements and methods for the People CMM-Based Assessment Method. This method is a diagnostic tool that supports, enables, and encourages an organization's commitment to improving its ability to attract, develop, motivate, organize, and retain the talent needed to steadily improve its organizational capability. The method helps an organization gain insight into its workforce capability by identifying strengths and weaknesses of its current practices related to the People CMM. The method focuses on identifying improvements that are most beneficial, given an organization's business goals and current maturity level. Brief descriptions of the method activities, roles, and responsibilities are provided. The SEI Appraiser Program is discussed, detailing the requirements for persons qualified to lead People CMM-Based Assessments.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr012/98tr012abstract.html

CMU/SEI-98-TR-011/
ADA351653

Agora: A Search Engine for Software Components

Seacord, R.; Hissam, S.; Wallnau, K.

Agora is a software prototype being developed by the Commercial Off-the-Shelf (COTS)-Based Systems Initiative at the Software Engineering Institute (SEI). The object of this work is to create an automatically generated and indexed worldwide database of software products classified by component model. Agora combines introspection with Web search engines to reduce the costs of bring software to, and finding components in, the software marketplace. This report describes Agora's role in an emerging component industry and the features and capabilities provided by Agora. The implementations of a JavaBeans agent and a Common Object Request Broker Architecture (CORBA) agent are also described. These agents are used to gather components of their respective types.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr011/98tr011abstract.html

CMU/SEI-98-TR-010
ADA351644

Browsers for Distributed Systems: Universal Paradigm or Siren's Song?

Seacord, R.; Hissam, S.

Web-based browsers are quickly becoming ubiquitous in the workplace. Software development managers are quick to incorporate browsers into a broad range of software development projects, often inappropriately. The purpose of this technical report is to examine the technical issues relevant to incorporating browsers as a component of a

commercial off-the-shelf (COTS) -based solution. Issues examined include portability, performance, functionality, severity, human factors, distribution, installation, upgrading, component-based development, runtime configuration management, and licensing.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr010/98tr010abstract.html

CMU/SEI-98-TR-009
ADA354756

Model-Based Verification: A Technology for Dependable Upgrade

Gluch, D. and Weinstock, C.

This is a preliminary report on the technological foundations of model-based verification for engineering software system upgrades. It describes the historical background and technical foundations for the approach and begins to provide a basis for the transition of model-based verification into practice. Critical technical and procedural issues that have been or are being addressed to ensure successful transition are examined. The report is aimed at providing technical insight and understanding for software management and engineering personnel on this emerging technology for verification of software system upgrades.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr009/98tr009abstract.html

CMU/SEI-98-TR-008
ADA350761

Architecture Tradeoff Analysis Method, The

Kazman, R.; Klein, M.; Barbacci, M.; Longstaff, T.; Lipson, H.; Carriere, J...

This paper presents the Architecture Tradeoff Analysis Method (ATAM), a structured technique for understanding the tradeoffs inherent in the architectures of software-intensive systems. This method was developed to provide a principled way to evaluate a software architecture's fitness with respect to multiple competing quality attributes: modifiability, security, performance, availability, and so forth. These attributes interact, and improving one often comes at the price of worsening one or more of the others. The method helps us reason about architectural decisions that affect quality attribute interactions. The ATAM is a spiral model of design, one of postulating candidate architectures followed by analysis and risk mitigation that lead to refined architectures.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr008/98tr008abstract.html

CMU/SEI-98-TR-007
ADA346252

DoD Product Line Practice Workshop Report

Bergey, J.; Clements, P.; Cohen, S.; Donohoe, P.; Jones, L.; Krut, B.; Northrop, L.; Tilley, S.; Smith, D.; Withey, J.

The Department of Defense (DoD) Product Line Practice workshop, Product Lines: Bridging the Gap—Commercial Success to DoD Practice was a hands-on meeting held in March 1998. Its purpose was to identify industry-wide best practices in software product lines, to share DoD product line experience, to explore the technical and non-technical issues involved, and to discuss ways in which the current gap between commercial best practice and DoD practice can be bridged. This report synthesizes the workshop presentations and discussions that described selected product line practices and identified barriers and enablers to achieving these practices within the DoD.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr007/98tr007abstract.html
Hughes Aircraft's Widespread Deployment of a

CMU/SEI-98-TR-006
ADA353168

Continuously Improving Software Process

Willis, R.; Rova, R.; Scott, M.; Johnson, M.; Ryskowski, J.; Moon, J.; Shumate, K.; Winfield, T. (Raytheon Systems Company)

This report describes the software improvement activities of Hughes Aircraft Company over the last 25 years. The focus is on continuous improvement of the software development process and the deployment of that process from a single organization at Fullerton, California, to virtually all the 5000 software engineers of Hughes Aircraft. For this achievement, the widespread deployment of a continuously improving software process, Hughes Aircraft was awarded the 1997 IEEE Computer Society Software Process Achievement Award.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr006/98tr006abstract.html

CMU/SEI-98-TR-005
ADA343688

Reverse-Engineering Environment Framework, A

Tilley, S.R.

This report describes a framework for reverse-engineering environments used to aid program understanding. The framework is based on a descriptive model that categorizes important support mechanism features based on a hierarchy of attributes. The attributes include cognitive model support, reverse-engineering tasks, canonical activities that are characteristic of the reverse-engineering process, quality attributes supported by the reverse-engineering environment, and miscellaneous characteristics.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr005/98tr005title.htm

CMU/SEI-98-TR-004
ADA353172

Transition Packages for Expediting Technology Adoption: The Prototype Requirements Management Transition Package

Fowler, P.; Patrick, M.

This report describes the experience of building and evaluating a prototype transition package for organizations implementing processes in support of the Requirements Management key process area of the Software Engineering Institute's Capability Maturity Model℠ for Software.1 This report presents our conclusions based on evaluation and review of the prototype by users typical of the audience targeted for transition packages. Feedback from these users indicated that they were typical "early or late majority" adopters. They found the transition package helpful for orientation and education as part of implementing requirements management practices in their organizations. This report also describes the foundations in research and practice on which the transition package concept is based. We argue in this report that transition packages, as part of a complete "whole product" that includes training and consulting, can be an effective mechanism for expediting the diffusion, adoption, and implementation of important technologies. Finally, we describe what we now know about creating transition packages and how they might be used.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr004/98tr004abstract.html

CMU/SEI-98-TR-003
ADA351640

Software Acquisition Improvement Framework (SAIF) Definition

Fisher, M.; Damer, R.; Reed, L.S.; Barbour, R.

The Software Acquisition Improvement Framework (SAIF) is a computer-aided system that supports the improvement of an organization's software acquisition process capability and performance. The framework integrates an acquisition-process reference model, such as the Software Acquisition Capability Maturity Model℠ (SA-CMM®); a process that defines the improvement approach, such as the SEI's IDEAL℠) method; plus guidance and other artifacts, which support the use of the model and improvement process. The guidance and artifacts are stored in a repository that automatically links them to the rest of the framework. This linking is structured to ensure that the reference model, the improvement process, and the supporting artifacts are available to the organization at

the right time in the improvement process phases and to focus on the areas for which the organization seeks improvement. This document discusses rationale behind the need for the SAIF, the elements constituting the SAIF, and the intended operational usage of the SAIF.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr003/98tr003abstract.html

CMU/SEI-98-TR-001
ADA340194

Coming Attractions in Program Understanding II: Highlights of 1997 and Opportunities in 1998

Tilley, S.R.

This report highlights important developments in program-understanding work in 1997 and outlines some of the opportunities in the field in 1998. A framework of three focus areas is used to categorize research and development activities in program understanding: investigating cognitive aspects, developing support mechanisms, and maturing the practice. Although significant progress was made in these areas, the rapid changes in the software engineering landscape are giving rise to several new challenges. Three of the most important in the coming year are leveraging the Web, black-box understanding, and the Year 2000 problem.

http://www.sei.cmu.edu/publications/documents/98.reports/98tr001/98tr001abstract.html

Special Reports

CMU/SEI-98-SR-013
ADA362584

Perceived Control of Software Developers and Its Impact on the Successful Diffusion of Information Technology

Green, G. (Baylor University); Hevner, A. (University of South Florida)

Why are beneficial software engineering practices not being used effectively in the development of software systems? This question has intrigued researchers in software engineering for many years. Billions of dollars per year are spent, and a large proportion wasted, on building and maintaining software systems that are either never completed or, if completed, are of poor quality. This state of software development has led to the introduction of innovative tools and techniques to support the software development process. Initial evidence from use of these tools and techniques shows significant improvements in development productivity and software quality. However, many of these potentially beneficial tools and techniques have not been widely adopted or diffused. This research seeks to examine the reason for why this is so: What factors explain the successful diffusion of new software development techniques into practice?

http://www.sei.cmu.edu/publications/documents/98.reports/98sr013/98sr013abstract.html

CMU/SEI-98-SR-006
ADA350855

Mapping MetaH into ACME

Barbacci, M.; Weinstock, C.

This report explores the translation of MetaH into ACME as a first step into the translation of MetaH to other architecture description languages (e.g., Rapide) to take advantage of any toolsets developed for the target language. We start by comparing the meta-models of ACME and MetaH, we establish mapping rules for each MetaH construct, and we present a full MetaH example taken from the MetaH Library at Honeywell. The report concludes with a brief description of possible alternative paths to obtain (limited) Rapide behavioral specifications from MetaH timing and sequencing of operations.

http://www.sei.cmu.edu/publications/documents/98.reports/98sr006/98sr006abstract.html

CMU/SEI-98-SR-003
ADA346343

Report on the Second International Workshop on Development and Evolution of Software Architectures for Product Families

Clements, P.C.; Weiderman, N.

In February 1998, the European Architectural Reasoning for Embedded Software (ARES) project sponsored the Second International Workshop on Development and Evolution of Software Architectures for Product Families. The workshop brought together practitioners and academics from Europe and the United States who are working in the area of software product families; that is, the production of related software systems from a common set of core assets. Chief among those assets is a shared software and/or system architecture. This workshop explored problems of architecture creation, description, evaluation, recovery, and architecture-based process in the context of building a product family. This report summarizes the discussions and outcomes of the workshop.

http://www.sei.cmu.edu/publications/documents/98.reports/98sr003/98sr003abstract.html

CMU/SEI-98-SR-002
ADA343704

Assessment of CORBA and POSIX Designs for FAA En Route Resectorization

Meyers, B.; Plakosh, D.; Place, P.; Klein, M.; Kazman, R.

Modernizing the En Route system presents major acquisition issues to the Federal Aviation Administration (FAA). At the present time, efforts are underway to upgrade the En Route system, primarily focusing on the host computer system. This report addresses the use of different technologies and an architectural tradeoff approach on a typical En Route system problem. We were requested to consider the problem of resectorization, i.e., the combination and decombination of sectors (and fix posting areas) during operation of the En Route center. Such capabilities may become desirable for an implementation of free flight. Two technologies have been applied to develop solutions to this problem, namely Common Object Request Broker Architecture (CORBA) and POSIX.21 (Portable Operating System Interface Standard). The former is based on an object-oriented model, while the latter is based on a message-passing model.

http://www.sei.cmu.edu/publications/documents/98.reports/98sr002/98sr002abstract.html

Handbooks

Handbook for Computer Security Incident Response Teams (CSIRTs)

West-Brown, M.; Stikvoort, D.; Kossakowski, K.

This document provides guidance on the generic issues to consider when forming and operating a computer security incident response team (CSIRT). In particular, it helps an organization to define and document the nature and scope of a computer security incident response (CSIR) service, which is the core service of a CSIRT. The document discusses the functions that make up the service; how those functions interrelate; and the tools, procedures, and roles necessary to implement the service. This document also describes how CSIRTs interact with other organizations and how to handle often sensitive information. In addition, operational and technical issues are addressed, such as equipment, security, and staffing considerations.

This document is intended to provide a valuable resource to both newly forming teams and existing teams whose services, policies, and procedures are not clearly defined or documented. The primary audience for this document consists of managers responsible for the creation or operation of a CSIRT or a CSIR service. It can also be used as a reference for all CSIRT staff, higher-level managers, and others who interact with a CSIRT.

http://www.sei.cmu.edu/publications/documents/98.reports/98hb001/98hb001abstract.html

Security Improvement Modules

CMU/SEI-SIM-005
ADA351646

Preparing to Detect Signs of Intrusion

Kochmar, J.; Allen, J.; Alberts, C.; Cohen, C.; Ford, G.; Fraser, B.; Konda, S.; Kossakowski, K.P.; Simmel, D.

It is essential that those responsible for your organization's information systems and networks be adequately prepared to detect evidence of breaches in security when they occur. Without advance preparation, it will be difficult, if not impossible, to determine if an intruder has been present and the extent of the damage caused by the intrusion.

Thorough preparation will permit you to detect an intrusion or an intrusion attempt during or soon after it occurs. Preparation involves consideration of your security policy and supporting procedures, your critical business information, your systems, your networks, your user community (internal and external), and the tools to be employed in detecting intrusions.

A general security goal is to prevent intrusions. Even if you have sophisticated prevention measures in place, your strategy for detecting intrusions must include preparation. This module is a companion to *Detecting Signs of Intrusion* (CMU/SEI-SIM-001).

The practices contained in this module identify advance preparations you must make to enable you to obtain evidence of an intrusion or an intrusion attempt. They are designed to help you prepare by configuring your data, systems, networks, workstations, tools, and user environments to capture the necessary information for detecting signs of intrusion.

http://www.cert.org/security-improvement/#modules

CMU/SEI-SIM-003
ADA336329

Security for Information Technology Service Contracts

Allen, J.; Ford, G.; Fraser, B.; Kochmar, J.; Konda, S.; Simmel, D.; Cunningham, L. (Computer Sciences Corporation)

An increasing number of organizations are organizations are contracting with outside companies for installation and maintenance of their information technology (IT). All too often, these organizations experience increased difficulty in providing appropriate oversight of the services and software for which they have contracted. For example, contractor access to the organization's systems is often neither well controlled nor secure, placing information systems and data at risk. The practices recommended in this document are designed to assist your organization in managing the contractor, managing the contract, and deterring common, known security problems when IT services and software are externally contracted.

http://www.cert.org/security-improvement/#modules

1997 Reports

Technical Reports

CMU/SEI-97-TR-029
ADA343692

Steps in an Architecture Tradeoff Analysis Method: Quality Attribute Models and Analysis

Barbacci, M.; Carriere, S.; Feiler, P.; Klein, M.; Lipson, H.; Longstaff, T.; Weinstock, C.

This paper presents some of the steps in an emerging architecture tradeoff analysis method (ATAM). The objective of the method is to provide a principled way to understand a software architecture's fitness with respect to multiple competing quality attributes: modifiability, security, performance, availability, and so forth. These attributes can interact or conflict--improving one often comes at the price of worsening one or more of the others, thus it is necessary to trade off among multiple software quality attributes at the time the software architecture of a system is specified, and before the system is developed. This report illustrates typical quality attribute models, analyses, and tradeoffs using a small real-time industrial application.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr029/97tr029abstract.html

CMU/SEI-97-TR-014
ADA336213

Approaches to Legacy System Evolution

Weiderman, N.H.; Smith, D.B.; Tilley, S.R.

The approach that one chooses to evolve software-intensive systems depends on the organization, the system, and the technology. We believe that significant progress in system architecture, system understanding, object technology, and net-centric computing make it possible to economically evolve software systems to a state in which they exhibit greater functionality and maintainability. In particular, interface technology, wrapping technology, and network technology are opening many opportunities to leverage existing software assets instead of scrapping them and starting over. But these promising technologies cannot be applied in a vacuum or without management understanding and control. There must be a framework in which to motivate the organization to understand its business opportunities, its application systems, and its road to an improved target system. This report outlines a comprehensive system evolution approach that incorporates an enterprise framework for the application of the promising technologies in the context of legacy systems.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr014/97tr014abstract.html

CMU/SEI-97-TR-013
ADA341963

Survivable Network Systems: An Emerging Discipline

Ellison, B.; Fisher, D.A.; Linger, R.C.; Lipson, H.F.; Longstaff, T.; Mead, N.R.

Society is growing increasingly dependent upon large-scale, highly distributed systems that operate in unbounded network environments. Unbounded networks, such as the Internet, have no central administrative control and no unified security policy. Furthermore, the number and nature of the nodes connected to such networks cannot be fully known. Despite the best efforts of security practitioners, no amount of system hardening can assure that a system that is connected to an unbounded network will be invulnerable to attack. The discipline of survivability can help ensure that such systems can deliver essential services and maintain essential properties such as integrity, confidentiality, and performance, despite the presence of intrusions. Unlike the traditional security measures that require central control or administration, survivability is intended to address unbounded network environments. This report describes the survivability approach to helping assure that a system that must operate in an unbounded network is robust in the presence of attack and will survive attacks that result in successful intrusions. Included

are discussions of survivability as an integrated engineering framework, the current state of survivability practice, the specification of survivability requirements, strategies for achieving survivability, and techniques and processes for analyzing survivability.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr013/97tr013abstract.html

CMU/SEI-97-TR-012
ADA331014

Discovering DISCOVER

Tilley, S.R.

This report describes investigations into DISCOVER, a modern software development and maintenance environment. The study is guided by a framework for classifying program understanding tools that is based on a description of the canonical activities that are characteristic of the reverse engineering process. Implications of this work for advanced practitioners, researchers and tool developers, and the framework itself are discussed.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr012/97tr012abstract.html

CMU/SEI-97-TR-011
ADA335653

Study in the Use of CORBA in Real-Time Settings: Model Problems for the Manufacturing Domain, A

Polze, A. (Humboldt University of Berlin); Plakosh, D.; Wallnau, K.

The Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA) is an important and popular technology that supports the development of object-based, distributed applications. The benefits promised by CORBA (abstraction, heterogeneity, etc.) are appealing in many application domains, including those that satisfy real-time requirements-such as manufacturing. Unfortunately, CORBA was not specified in light of real-time requirements, and so the question remains whether existing object request brokers (ORBs) can be used in real-time settings, or whether developers of real-time systems must await future extensions of CORBA that address real-time issues or use non-CORBA- compliant ORBs. In this report, we describe the application of an off-the-shelf ORB to two real-time model problems. Based on our experiences, we believe that today's ORBs can be used in real-time settings, with certain caveats as outlined in this report. We also outline the concept of composite objects, an approach for extending the range of non-real-time ORBs into a greater variety of real-time settings.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr011/97tr011abstract.html

CMU/SEI-97-TR-010
ADA330928

Playing Detective: Reconstructing Software Architecture from Available Evidence

Kazman, R.; Carriere S.J.

Because a system's software architecture strongly influences its ability to support quality attributes such as modifiability, performance, and security, it is important to be able to analyze and reason about that architecture. However, architectural documentation frequently does not exist, and when it does, it is often out of sync with the implemented system. In addition, it is rare that software development begins with a clean slate; systems are almost always constrained by existing legacy code. As a consequence, we need to be able to extract information from existing system implementations and reason architecturally about this information. This paper presents Dali, an open, lightweight workbench that aids an analyst in extracting, manipulating, and interpreting architectural information. By assisting in the reconstruction of architectures from extracted information, Dali helps an analyst redocument architectures and discover the relationship between "as-implemented" and "as-designed" architectures.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr010/97tr010abstract.html

CMU/SEI-97-TR-009
ADA350658

Approach for Selecting and Specifying Tools for Information Survivability, An

Firth, R.; Fraser, B.; Konda, S.; Simmel, D.

As today's technology becomes increasingly complex to manage, administrators of survivable systems will need to place increased reliance on tools to assist them. The selection and specification of these tools must be conducted in a reliable, systematic fashion. This paper proposes a lexicon of functionalities to characterize survivable systems activities, and an approach to analyze networked systems environments. Application of this analysis approach will assist organizations in establishing criteria for selecting tools, and to identifying requirements for new tool development to accommodate needs not met by currently available tools.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr009/97tr009abstract.html

CMU/SEI-97-TR-008
ADA331480

Software Process Automation: Interviews, Survey, and Workshop Results

Christie, A.; Levine, L.; Morris, E.J.; Riddle, B.; Zubrow, D. (Software Engineering Institute)
Belton, T.; Proctor, L. (Nolan Norton and Company)
Cordelle, D.; Ferotin, J. Solvay, J. (Cap Gemini Segoti)

This report describes the results of a two-year study of experiences with the adoption and use of software process automation. The work was motivated by a desire to provide insights and guidelines to those planning to implement this technology. The focus of the study was primarily, but not exclusively, on end-user organizations. The study was conducted in three stages: First, in-depth interviews were conducted to assess the state of the practice. Second, a survey questionnaire was distributed to a wider number of organizations to obtain more quantitative data. The populations in these two groups turned out to be quite different, a fact that we believe enriches the content of this report. Finally, a one-day workshop was held, the objective of which was to explore with practitioners why the gap between the theory and practice of software process automation is as large as it is. A previous report by Alan Christie, et al. [Christie 96] documented the results of the in-depth interviews in detail. This report now summarizes the results of the interviews, and describes in more detail the questionnaire survey and the workshop. It also provides both insight for process automation tool developers and guidelines for adoption to process-automation end users.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr008/97tr008abstract.html

CMU/SEI-97-TR-007
ADA330880

Enterprise Framework for the Disciplined Evolution of Legacy Systems

Bergey, J.K.; Northrop, L.M.; Smith, D.B.

Many organizations are planning to "migrate" their legacy systems to distributed open system environments or a single product line of systems. Many of these efforts are often less than successful because they concentrate on a narrow set of software issues without fully considering a broader set of enterprise-wide management and technical issues. This report describes an enterprise framework that characterizes the global environment in which system evolution takes place and provides insight into the activities, processes, and work products that shape the disciplined evolution of legacy systems. Exemplary checklists are included to identify critical enterprise issues corresponding to each of the framework's elements. Preliminary results indicate that the enterprise model is a useful tool for probing and evaluating planned and ongoing system evolution initiatives. The model serves to draw out important global issues early in the planning cycle and provides insight for developing a synergistic set of management and technical practices to achieve a disciplined approach to system evolution.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr007/97tr007abstract.html

CMU/SEI-97-TR-005
ADA326945

Implications of Distributed Object Technology for Reengineering

Weiderman, N.; Northrop, L.; Smith, D.; Tilley, S.; Wallnau, K.

Distributed object technology is profoundly changing the ways in which software systems evolve over time. To a large extent, the focus of reengineering has been to understand legacy systems and to extract their essential functionality so that they can be rewritten as more robust and more maintainable systems over the long term. However, object technology, wrapping strategies, and the Web may be changing the focus and economics of reengineering. The question posed by this paper is the extent to which reengineering strategies ought to continue to use program understanding technology. The cost/benefit ratio of certain forms of program understanding appears

to be staying roughly the same over time, while the cost/benefit ratio of wrapping legacy systems or their subsystems is dropping rapidly. As a result, new reengineering strategies that place less emphasis on deep program understanding, and more emphasis on distributed object technologies, should now be considered.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr005/97tr005abstract.html

CMU/SEI-97-TR-004
ADA327035

Distributed Object Technology with CORBA and Java: Key Concepts and Implications

Wallnau, K.; Weiderman, N.; Northrop, L.

The purpose of this report is to analyze the potential impact of distributed object technology (DOT) on software engineering practice. The analysis culminates with the conclusion that the technology will have a significant influence on both the design and reengineering of information systems and the processes used to build them. We see a profound impact and fundamental change in both technical thinking and practice as a result of the related technologies we group together as DOT.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr004/97tr004abstract.html

CMU/SEI-97-TR-003
ADA327610

Product Line Practice Workshop Report

Bass, L.; Clements, P.; Cohen, S.; Northrop, L.; Withey, J.

The first Software Engineering Institute Product Line Practice Workshop was a hands-on meeting held in December 1996 to share industry and government practices in software product lines and to explore the technical and non-technical issues involved. This report synthesizes the workshop presentations and discussions, which identified factors involved in product line practices and analyzed issues in the areas of architecture, people-organization-management, and business models.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr003/97tr003abstract.html

CMU/SEI-97-TR-002
ADA325361

Year 2000 Problem: Issues and Implications, The

Smith, D.; Muller, H.; Tilley, S.

A lot of attention has recently focused on the possibility that a great deal of software will fail at the turn of the century because of the way dates are stored and processed by computer programs. Attitudes range from alarmist to unconcerned regarding the magnitude and implications of the problem. This report outlines the basic issues of the so-called “Year 2000" (Y2K) problem and discusses some of its implications.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr002/97tr002abstract.html

CMU/SEI-97-TR-001
ADA335543

Personal Software Process (PSP): An Empirical Study of the Impact of PSP on Individual Engineers, The

Hayes, W.; Over, J.W.

This report documents the results of a study that is important to everyone who manages or develops software. The study examines the impact of the Personal Software Process (PSP) on the performance of 298 software engineers. The report describes the effect of PSP on key performance dimensions of these engineers, including their ability to estimate and plan their work, the quality of the software they produced, the quality of their work process, and their productivity. The report also discusses how improvements in personal capability also improve organizational performance in several areas: cost and schedule management, delivered product quality, and product cycle time.

http://www.sei.cmu.edu/publications/documents/97.reports/97tr001/97tr001abstract.html

Special Reports

CMU/SEI-97-SR-019
ADA332592

Workshop on COTS-Based Systems

Oberndorf, P.; Brownsword, L.; Morris, E.; Sledge, C.

This report documents the proceedings of the first Workshop on COTS-Based Systems, held at the Software Engineering Institute (SEI) June 10-11, 1997. It describes the workshop activities, the discussions of the three breakout groups, and some general conclusions reached by the participants in the workshop.

http://www.sei.cmu.edu/publications/documents/97.reports/97sr019/97sr019abstract.html

CMU/SEI-97-SR-018
ADA332483

Directory of Industry and University Collaborations with a Focus on Software Engineering Education and Training, Version 6

Beckman, K. (Computer Data Systems, Inc.)

This directory describes 24 formal collaborative efforts to promote software engineering education and training activities among industry organizations (including government) and universities in the United States, Canada, and Australia. These collaborations vary in their organizational structure and types of services offered. All attempt to bridge the gap between industry needs and academic software engineering education and training offerings. Readers can use this directory to find collaborations that match their software engineering education and training needs and that are located within their geographic area.

CMU/SEI-97-SR-016
ADA330926

Report of the STEP '97 Workshop on Net-Centric Computing

Tilley, S.R.; Storey, M.D.

As part of the STEP '97 conference, workshop W2b, Net-Centric Computing, was held on Thursday, July 17, 1997. Through focused presentations and open discussion, this workshop explored net-centric computing and its potential impact on software users, application developers, and system administrators. This report describes the STEP '97 conference, overviews the Net-Centric Computing workshop, and provides a summary of the invited presentations.

http://www.sei.cmu.edu/publications/documents/97.reports/97sr016/97sr016abstract.html

CMU/SEI-97-SR-014
ADA329326

Workshop on the State of the Practice in Dependably Upgrading Critical Systems

Gluch, D.P.; Weinstock, C.B.

This report describes the results of the Workshop on the State of the Practice in Dependably Upgrading Critical Systems held April 16-17, 1997 at the Software Engineering Institute. The workshop addressed a broad spectrum of issues associated with dependably and cost-effectively upgrading systems, primarily those with reliability or real-time requirements.

CMU/SEI-97-SR-013
ADA328632

Software Acquisition Process Maturity Questionnaire

Ferguson, J.; Cooper, J.; Falat, M.; Fisher, M.; Guido, A.; Marciniak, J.; Matejceck, J.; Webster, R.

This package contains a copy of the software acquisition process maturity questionnaire. It is intended for those interested in performing and learning about software acquisition process appraisals. This questionnaire is not an appraisal method itself; rather, it is a tool that may be used in different appraisal methods.

http://www.sei.cmu.edu/publications/documents/97.reports/97sr013/97sr013abstract.html

CMU/SEI-97-SR-010
ADA328634

Report of the Reuse and Product Lines Working Group of WISR8

Clements, P.

This report summarizes the discussions held by the Reuse and Product Lines working group at the Eighth Workshop on Software Reuse (WISR8). The working group was chartered to explore the range of issues and practices necessary for the successful fielding of a software product line, which is a collection of systems that are built from a common set of core assets. Issues addressed include the relation between a product line and a domain, benefits of the product line approach to software development, organizational factors, effects of the products' domain and context, risks, the role of architecture and architects, and managing change and variation. Maintaining intellectual control was a theme that arose repeatedly during the discussion.

http://www.sei.cmu.edu/publications/documents/97.reports/97sr010/97sr010abstract.html

CMU/SEI-97-SR-009
ADA331515

How to Use the Software Process Framework

Gates, L.P.

This report is intended to provide guidance on how to use the Software Process Framework (SPF) [Olson 94] for reviewing, analyzing, and designing software process documents that are consistent with the *Capability Maturity Model*[SM] *(CMM®) for Software, Version 1.1* [Paulk 93a]. This guidance is not "how to design" or "how to analyze" software process documents in general. Rather, the guidance is focused on how to use the Software Process Framework for those purposes. The purpose of this report is to clarify the intended usage of the SPF and describe usage scenarios that have evolved through the use of the SPF in the software development community over several years. This report is intended to be used as a supplement to the SPF, not by itself. It is assumed that the reader is familiar with the CMM, is experienced in software process improvement and definition, and has skill in designing or analyzing software process documents.

http://www.sei.cmu.edu/publications/documents/97.reports/97sr009/97sr009abstract.html

CMU/SEI-97-SR-008
ADA327776

Perspective on the State of Research in Fault-Tolerant Systems, A

Weinstock, C.; Gluch, D.

As computers take on a greater role in society, their dependability is becoming increasingly important. Given software's critical role in computing systems, reliable software has emerged as crucial to achieving a dependable infrastructure. Using a system perspective that recognizes the prominence of software, we characterize the current state of fault-tolerance research as it contributes to the dependability of computer systems and we conjecture on future directions for this research area.

http://www.sei.cmu.edu/publications/documents/97.reports/97sr008/97sr008abstract.html

CMU/SEI-97-SR-003
ADA324232

Report to the President's Commission on Critical Infrastructure Protection

Ellis, J.; Fisher, D.; Longstaff, T.; Pesante, L.; Pethia, R.

This report was submitted to the President's Commission on Critical Infrastructure Protection for their consideration. Based on the experience of the CERT[SM] Coordination Center, we identify threats to and vulnerabilities of the Internet and estimate the cascade effect that a successful, sustained attack on the Internet would have on the critical national infrastructures set out in Executive Order 13010. Finally, we discuss the implications for public policy and make specific recommendations.

http://www.sei.cmu.edu/publications/documents/97.reports/97sr003/97sr003abstract.html

CMU/SEI-97-SR-002
ADA324230

Turbo-Team Approach to Establishing a Software Test Process at Union Switch & Signal, A

McAndrews, D.; Marchok Ryan, J.; Fowler, P.

Process improvement teams are often created and tasked to make complex changes in an organization, but are not provided with the tools necessary for success. This report describes what one team did to successfully install a software testing process in an organization. The principles of a turbo-team approach are introduced and a defined process for working in teams to introduce a new software technology is adapted from the prototype Process Change Guide developed by the Software Engineering Institute (SEI). Artifacts of this effort are included and described as examples for other teams to reference. Successes and lessons learned are described.

http://www.sei.cmu.edu/publications/documents/97.reports/97sr002/97sr002abstract.html

CMU/SEI-97-SR-001
ADA325553

Proceedings of the Introducing Requirements Management into Organizations Workshop: Requirements Management Transition Packages (November 11-13, 1996)

Fowler, P.; Patrick, M.

This document summarizes the findings and presents the raw data from the Introducing Requirements Management into Organizations workshop hosted by the SEI (Software Engineering Institute) in November 1996. A transition package consists of a process description, related materials for users of the description, and materials for use by change agents in action teams and technical working groups introducing requirements management processes and tools into their organizations. The workshop participants considered the feasibility of building a transition package to expedite the adoption of effective requirements management practice and concluded that a transition package can and should be built for requirements management. This document records and publicizes the findings of the workshop, including problems and opportunities related to requirements management transition packages identified by workshop participants.

http://www.sei.cmu.edu/publications/documents/97.reports/97sr001/97sr001abstract.html

Security Improvement Modules

CMU/SEI-SIM-001
ADA329629

Detecting Signs of Intrusion

Firth, R.; Ford, G; Fraser, B.; Kochmar, J.; Konda, S; Richael, J.; Simmel, D. (Networked Systems Survivability Program); Cunningham, L. (Computer Sciences Corporation)

The module provides concrete, practical guidance to help organizations improve the security of their networked computer systems. It describes a set of practices that can help detect intrusions by looking for the “fingerprints” of known intrusion methods.

http://www.cert.org/security-improvement/#modules

Handbooks

CMU/SEI-97-HB-003
ADA325551

Practical Software Measurement: Measuring for Process Management and Improvement

Florac, W.; Park, R.; Carleton, A.

This guidebook shows how well-established principles and methods for evaluating and controlling process performance can be applied in software settings to help achieve an organization's business and technical goals. Although the concepts that are illustrated are often applicable to individual projects, the primary focus is on the enduring issues that enable organizations to improve not just today's performance, but the long-term success and profitability of their business and technical endeavors.

http://www.sei.cmu.edu/publications/documents/97.reports/97hb003/97hb003abstract.html

CMU/SEI-97-HB-002
ADA328098

Software Acquisition Risk Management Key Process Area (KPA)—A Guidebook Version 1.0

Gallagher, B.; Alberts, C.; Barbour, R.

In this guidebook, we hope to provide sponsors of acquisition improvement programs and their immediate staff with guidelines on how to implement a software acquisition risk management program satisfying the goals of the Acquisition Risk Management (ARM) Key Process Area (KPA) of the Software Acquisition Capability Maturity Model (SA-CMM). Brief overviews of software acquisition and the SA-CMM are included.

http://www.sei.cmu.edu/publications/documents/97.reports/97hb002/97hb002abstract.html

CMU/SEI-97-HB-001
ADA320732

C4 Software Technology Reference Guide—A Prototype

Foreman, J.; Gross, J.; Rosenstein, R.; Fisher, D.; Brune, K.

The Air Force acquisition community tasked the Software Engineering Institute (SEI) to create a reference document that would provide the Air Force with a better understanding of software technologies. This knowledge will allow the Air Force to systematically plan the research and development (R&D) and technology insertion required to meet current and future Air Force needs, from the upgrade and evolution of current systems to the development of new systems.

The initial release of the Software Technology Reference Guide is a prototype to provide initial capability, show the feasibility, and examine the usability of such a document. This prototype generally emphasizes software technology of importance to the C4I (command, control, communications, computers, and intelligence) domain. This emphasis on C4I neither narrowed nor broadened the scope of the document; it did, however, provide guidance in seeking out requirements and technologies. It served as a reminder that this work is concerned with complex, large-scale, distributed, real-time, software-intensive, embedded systems in which reliability, availability, safety, security, performance, maintainability, and cost are major concerns.

http://www.sei.cmu.edu/activities/str/index.html
http://www.sei.cmu.edu/publications/documents/97.reports/97hb001/97hb001abstract.html

Guidebooks

Continuous Risk Management Guidebook

Dorofee, A.J.; Walker, J.A.; Alberts, C.J.; Higuera, R.P.; Murphy, R.L.; Williams, R.C.

The purpose of the Continuous Risk Management Guidebook is to explain what Continuous Risk Management is; to help you understand the principles, functions, methods, and tools; to show what it could look like when implement its own adaptation. The intent is not to provide a "cookie-cutter" answer for everyone. There is no such answer. This is a generic practice with a variety of methods and tools from which to choose. It is meant to be adapted to suit an organization and a project.

To purchase the guidebook, please contact-

Michelle Woods
Center for Information Systems Engineering
Carnegie Mellon University
(412) 268-6322
FAX (412) 268-6369

Prices: $70 for industry and academia
$56 for government: no additional charge for shipping and handlingS

1996 Reports

Technical Reports

CMU/SEI-96-TR-036
ADA324233

Principles for Evaluating the Quality Attributes of a Software Architecture

Barbacci, M.; Klein, M.; Weinstock, C.

Software quality is the degree to which software possesses a desired combination of attributes (e.g., reliability, interoperability). In this paper we describe a few principles for analyzing a software architecture to determine if it exhibits certain quality attributes. We show how analysis techniques indigenous to the various quality attribute communities can provide a foundation for performing software architecture evaluation. We also show how the principles provide a context for existing evaluation approaches such as scenarios, questionnaires, checklists, and measurements. Our immediate goal in identifying these principles for attribute-based architecture evaluation is to better integrate existing techniques and metrics into software architecture practice, not necessarily to invent new attribute-specific techniques and metrics. A longer-term goal is to codify these principles into systematic procedures or methods for architecture evaluation. This paper is an initial step towards identifying the ingredients of such methods.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.036.html

CMU/SEI-96-TR-035
ADA320528

Case Study in Structural Modeling, A

Chastek, G.; Brownsword, L.

This report is one in a series of Software Engineering case studies in software architecture. It describes structural modeling, a technique for creating software architectures based on a small set of design elements called structural types. Structural modeling resulted from the efforts of the Air Force Aeronautical Systems Command (ASC/YW) and has been used by Air Force contractors since the late 1980s to design large-scale, high-fidelity aircrew trainer simulation software. This report examines the changes, resulting from the use of structural modeling, to the trainer's software architecture and to the development methods used.

CMU/SEI-96-TR-034
ADA324517

Best Training Practices Within the Software Engineering Industry

Mead, N.; Tobin, L.; Couturiaux, S.

This report provides the results of a benchmarking study to identify the best training practices within the software engineering community. We surveyed 24 organizations to create a broad picture of training as it currently exists in industry. We then chose three of these organizations for an in-depth study to identify the best training practices and enablers to those practices. This report summarizes the results of the survey and the in-depth study, and discusses the best practices and enablers that were identified.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.034.html

CMU/SEI-96-TR-025
ADA320786

Recommended Best Industrial Practice for Software Architecture Evaluation

Abowd, G.; Bass, L.; Clements, P.; Kazman, R.; Northrop, L.; Zaremski, A.

Architectural decisions have a great impact on the consequent quality of software systems. As a result, it is important to evaluate how a software architecture meets it quality demands. Though much focus has been placed on modeling and describing the software architecture as a design artifact, we found that relatively little is known about the current experience with software architecture.

This report details the results of two workshops on software architecture evaluation, held at the Software Engineering Institute (SEI) on November 9-10, 1995 and May 9-10, 1996. The purpose of the workshops was to determine the state of industrial practice in the evaluation of software architectures with respect to a set of desired quality attributes, and to uncover recommendations for best practices. In this report, we summarize the findings of the two workshops, define a set of dimensions to characterize various software architecture evaluation techniques, and make concrete recommendations for implementing architecture evaluation practices.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.025.html

CMU/SEI-96-TR-024
ADA327609

Radical Improvements Require Radical Actions: Simulating a High-Maturity Software Organization

Burke, S. (CSC Resident Affiliate, Process Program)

This report describes the methodology used to create a simulation of a high-maturity software organization and the results of that simulation. The goal of this research was to find the quantitative value of improving from the Capability Maturity Model^{SM} (CMM^{SM}) Level 3 to Level 5. The method was to simulate a high-maturity organization using its actual empirical data and then "cut out" the high-maturity elements of the simulation. The resulting change in software size, effort, schedule, and quality would be a more accurate measure of the value of high maturity than working forward with a low- or medium-maturity organization and merely hypothesizing the activities and values of high maturity. The author used computer simulations based on systems thinking and systems dynamics, which reasonably modeled the "soft variables" of the people aspects of an organization (personnel attitudes, learning curve, participation in software process improvement, etc.). The simulation also related the soft variables to the hard variables of a software organization's life-cycle process (software size, effort, schedule, quality, etc.).

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.024.html

CMU/SEI-96-TR-023
ADA320485

Cleanroom Software Engineering Implementation of the Capability Maturity Model (CMM^{SM}) for Software

Linger, R.; Paulk, M.; Trammel, C. (University of Tennessee)

The Capability Maturity Model^{SM} for Software (CMM^{SM}) [CMU 95] developed by the Software Engineering Institute, and Cleanroom Software Engineering [Mills 87, Linger 93, Linger 94] developed by Dr. Harlan Mills and his associates in IBM and other organizations, share a common concern with software quality and the effectiveness of software development. The principal focus of the CMM is on process management maturity; the principal focus of Cleanroom is on rigorous engineering processes. The CMM management processes and the Cleanroom engineering processes are complementary and mutually reinforcing.

The Capability Maturity Model for Software describes the principles and practices underlying software process maturity. It is intended to help software organizations improve the maturity of their software processes through an evolutionary path from ad hoc, chaotic processes to mature, disciplined software processes. The CMM is organized into five maturity levels. The maturity levels are defined in terms of 18 key process areas (KPAs) that characterize project performance at each level.

Cleanroom software engineering is a theory-based engineering process for development and certification of high-reliability software systems under statistical quality control. Cleanroom is intended to help software organizations improve their ability to apply engineering discipline to software development. Cleanroom is defined in terms of 14 processes that implement the technology and operations involved in Cleanroom software development.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.023.html

CMU/SEI-96-TR-022
ADA319071

Cleanroom Software Engineering Reference

Linger, R.C.; Trammell, C.J.

Cleanroom software engineering is a theory-based team-oriented process for development and certification of high-reliability software systems under statistical quality control [Mills 92, Linger 93, Linger 94]. A principal objective of the Cleanroom process is development of software that exhibits zero failures in use. The Cleanroom name is borrowed from hardware Cleanrooms, with their emphasis on rigorous engineering discipline and focus on defect prevention rather than defect removal. Cleanroom combines mathematically based methods of software specification, design, and correctness verification with statistical, usage-based testing to certify software fitness for use. Cleanroom projects have reported substantial gains in quality and productivity.

This report defines the Cleanroom Software Engineering Reference Model, or CRM. The CRM is expressed in terms of a set of 14 Cleanroom processes and 20 work products. It is intended as a guide for Cleanroom project management and performance, process assessment and improvement, and technology transfer and adoption.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.022.html

CMU/SEI-96-TR-020
ADA320606

Software Acquisition Capability Maturity Model

Ferguson, J.; Cooper, J.; Falat, M.; Fisher, M.; Guido, A.; Marciniak, J.; Webster, R.

The SA-CMM[SM] is a model for benchmarking and improving the software acquisition process. The model follows the same architecture as the Capability Maturity Model for Software (SW-CMM), but with a unique emphasis on acquisition issues and the needs of individuals and groups who are planning and managing software acquisition efforts.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.020.html

CMU/SEI-96-TR-019
ADA320731

Coming Attractions in Program Understanding

Tilley, S.; Smith, D.

Program understanding is the (ill-defined) deductive process of acquiring knowledge about a software artifact through analysis, abstraction, and generalization. This report identifies some of the emerging technologies in program understanding. We present technical capabilities currently under development that may be of significant benefit to practitioners within five years. Three areas of work are explored: investigating cognitive aspects, developing support mechanisms, and maturing the practice.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.019.html

CMU/SEI-96-TR-018
ADA313952

Concept of Operations for ESC's Product Line Approach

Cohen, S.; Friedman, S.; Martin, L.; Royer, T.; Solderitsch, N.; Webster, R.

This document describes the Concept of Operations (ConOps) and transition strategy for the product line approach to software systems development at the Air Force Electronic Systems Center (ESC), Hanscom Air Force Base, Massachusetts. This document defines the organizations implementing the product line approach and the processes used by the organizations. It describes the roles and responsibilities, and the relationships among the organizations. The processes needed to implement software system development using the product line approach are described at a high level and will be further detailed and refined as the product line approach is implemented at ESC.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.018.html

CMU/SEI-96-TR-017
ADA317090

Transitioning a Model-Based Software Engineering Architectural Style to Ada 95

Gargaro, A.; Peterson, A.S.

This report describes the transition of an existing model-based software engineering architectural style to Ada 95. The report presents an overview of software architecture for developing product families of domain-specific applications comprising reusable components, explains recognized deficiencies in the existing Ada mapping to this software architecture, and proposes solutions for correcting these deficiencies using a mapping to Ada 95. The report concludes with observations gained during the transition exercise and recommendations for future activities aimed towards deploying and enhancing the proposed mapping.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.017.html

CMU/SEI-96-TR-016
ADA315802

Case Study in Successful Product Line Development, A

Brownsword, L.; Clements, P.

A product line is a set of related systems that address a market segment. Building a product line out of a common set of core assets, as opposed to building each member system separately, epitomizes reuse. Although software technology is key to achieving a product line capability, organizational and process considerations are just as crucial. This report describes the experience of one company, CelsiusTech Systems AB of Sweden, that builds large, complex, embedded, real-time shipboard command-and-control systems as a product line, developed in common from a base set of core software and organizational assets. The report describes the changes that CelsiusTech had to make to its software, organizational, and process structures to redirect the company toward a product line approach that yielded substantial economic and marketplace benefits to the company.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.016.html

CMU/SEI-96-TR-014
ADA310912

Controlled Experiment Measuring the Effect of Procedure Argument Type Checking on Programmer Productivity, A

Prechelt, L.; Tichy, W

Type checking is considered an important mechanism for detecting programming errors, especially interface errors. This report describes an experiment to assess the error-detection capabilities of static intermodule type checking.

The experiment uses ANSI C and Kernighan&Ritchie (K&R) C. The relevant difference is that the ANSI C compiler checks module interfaces (i.e., the parameter lists of calls to external functions), whereas K&R C does not. The experiment employs a counterbalanced design in which each subject writes two non-trivial programs that interface with a complex library (Motif). Each subject writes one program in ANSI C and one in K&R C. The input to each compiler run is saved and manually analyzed for errors.

Results indicate that delivered ANSI C programs contain significantly fewer interface errors than delivered K&R C programs. Furthermore, after subjects have gained some familiarity with the interface they are using, ANSI C programmers remove errors faster and are more productive (measured in both time to completion and functionality implemented).

This report describes the design, setup, and results of the experiment including complete source code and error lists.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.014.html

CMU/SEI-96-TR-013
ADA310916

Software Process Automation: Experiences from the Trenches

Christie, A.; Levine, L.; Morris, E.; Zubrow, D.; Belton, T.; Proctor, L.; Cordelle, D.; Ferotin, J.; Solvay, J.; Segoti, G.

Software process automation is a new technology with significant promise. However, practical experience in the field is still limited and there appears to be a variety of potential barriers to its use. The objective of this empirical study is to document current practical experience and to identify what works and what does not. Lessons learned

from the study will be disseminated to help others who wish to implement the technology. This report documents results from the first phase of the study in which 14 in-depth interviews were conducted. Personnel interviewed were involved in projects in which process-centered environments were developed and adopted.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.013.html

CMU/SEI-96-TR-012
ADA315789

Software Risk Management

Higuera, R.; Haimes, Y.

This paper presents a holistic vision of the risk-based methodologies for Software Risk Management (SRM) developed at the Software Engineering Institute (SEI). SRM methodologies address the entire life cycle of software acquisition, development, and maintenance. This paper is driven by the premise that the ultimate efficacy of the developed methodologies and tools for software engineering is to buy smarter, manager more effectively, identify opportunities for compatible information and databases more efficiently, improve industry raise the community's playing field, and review and evaluate progress. The methodologies are based on seven management principles: shared product vision, teamwork, global perspective, forward-looking view, open communication, integrated management, and continuous process.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.012.html

CMU/SEI-96-TR-010
ADA315653

Investment Analysis of Software Assets for Product Lines

Withey, J.

Group, product line, and program managers are faced with of allocating resources to projects. Should all resources be dedicated to meet near-term deliverables? Or should some be siphoned off to build software assets that may improve quality, flexibility, and reduce cost and time-to-market of future products in the product line? These managers also have to determine which assets to buy or build. The choices are many, ranging from reusable code components to design models to application generators, and each has a different risk and cash flow profile.

This report introduces an approach that will help managers make these allocation decisions. The report outlines a planning and communication tool for analyzing investments in software assets for product lines.

Although the report is not a guidebook, the concepts, criteria, and investment modeling techniques will be useful in making and justifying proposals for funding. The concepts are drawn from the fields of microeconomics, corporate finance, marketing, R&D technology management, and software reuse.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.010.html

CMU/SEI-96-TR-009
ADA310918

Transitioning Domain Analysis: An Industry Experience

Schnell, K.; Zalman, N

This report provides an industry experience in the planning and execution of a research project to pilot the use of the Software Engineering Institute (SEI) (domain analysis methodology known as feature-oriented domain analysis (FODA). Supported by examples, experiences, and lessons learned from the industry pilot study conducted by Nortel in collaboration with the SEI, this report addresses seven key areas: (1) Nortel's motivation for change; (2) defining the problem area and the search for a range of solution possibilities and/or approaches; (3) obtaining sponsorship, participants, and funding; (4) development of the project plan and contract; (5) implementation of the project plan for the pilot study; (6) completion and closure of the pilot study; and (7) the transition and deployment of FODA.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.009.html

CMU/SEI-96-TR-008
ADA309156

Coming Attractions in Software Architecture

Clements, P.

Software architecture is a field of study enjoying unprecedented growth and interest. This report identifies a set of promising lines of research related to software architecture and architecture-based system development that are expected to lead to advances available soon to practitioners. Some of the goals of software architecture are enumerated, and the investigatory efforts are structured according to work in the design or selection and creation, representation, evaluation and analysis, utilization, or legacy recovery of software architectures. Promising research is described in each area. These opinions are correlated with those of other experts in the field. Finally, a timeline for achieving some of the predicted results is offered.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.008.html

CMU/SEI-96-TR-007
ADA307934

CMM^{SM}-Based Appraisal for Internal Process Improvement (CBA IPI): Method Description

Dunaway, D.; Masters, S.

This document is a high-level overview of the CMM^{SM}-Based Appraisal for Internal Process Improvement (CBA IPI) V1.1 assessment method. It provides a brief history of SEI appraisal methods, as well as establishing appraisals in the context of the IDEAL^{SM} approach to software process improvement. CBA IPI is a diagnostics, enables, and encourages an organization's commitment to process improvement. The method helps an organization gain insight into its software development capability by identifying strengths and weaknesses of its current processes related tot he Capability Maturity Model^{SM} for Software V1.1. The method focuses on identifying software improvements that are organization's business goals and current maturity level. Brief descriptions of the method activities, roles, and responsibilities are provided. In addition, guidelines are provided for establishing resource requirements for conducting a CBA IPI. The SEI Appraiser Program is discussed, detailing the requirements for persons qualified to lead CBA IPIs.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.007.html

CMU/SEI-96-TR-006
ADA307890

Architectural Description of the Simplex Architecture, An

Rivera, J.; Danylyszyn, A.; Weinstock, C.; Sha, L.; Gagliardi, M.

Simplex is a software architecture for dependable and evolvable process-control systems developed by the Software Engineering Institute. Our project consisted of creating a formal specification of this architecture, and analyzing its safety and liveness properties. We developed a Communicating Sequential Processes (CSP) model to describe the overall dynamic behavior of the Simplex architecture, which we verified using the Failure-Divergence Refinement (FDR) model checker. As a result, we discovered interesting things about the use of FDR that revealed subtle points in the Simplex architecture. We also developed a WRIGHT specification of this architecture to characterize precisely the connections between its components at the architectural level. The specification was based on the latest version of the CSP model.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.006.html

CMU/SEI-96-TR-004
ADA307889

Mature Profession of Software Engineering, A

Ford, G.; Gibbs, N.

A model is presented that allows the characterization of the maturity of a profession in terms of eight infrastructure components: initial professional education, accreditation, skills development, certification, licensing, professional development, a code of ethics, and a professional society. Several mature professions are examined to provide examples of the nature of these components. The current states of the components of software engineering are described, and predictions are made for the evolution of those components as the profession matures.

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.004.html

CMU/SEI-96-TR-003
ADA305470

Software Architecture: An Executive Overview

Clements, P., Northrop, L.

Software architecture is an area of growing importance to practitioners and researchers in government, industry, and academia. Journals and international workshops are devoted to it. Working groups are formed to study it. Textbooks are emerging about it. The government is investing in the development of software architectures as core products in their own right. Industry is marketing architectural frameworks such as CORBA. Why all the interest and investment? What is software architecture, and why is it perceived as providing a solution to the inherent difficulty in designing and developing large, complex systems? This report will attempt to summarize the concept of software architecture for an intended audience of mid to senior level management. The reader is presumed to have some familiarity with common software engineering terms and concepts, but not to have a deep background in the field. This report is not intended to be overly-scholarly, nor is it intended to provide the technical depth necessary for practitioners and technologists. The intent is to distill some of the technical detail and provide a high level overview.

CMU/SEI-96-TR-002
ADA309160

Software Capability Evaluation Version 3.0 Method Description

Byrnes, P.; Phillips, M.

This report describes Version 3.0 of the Software Capability Evaluation (SCE) Method. SCE is a method for evaluating the software process of an organization to gain insight into its process capability. This version of the SCE Method is based on the Capability Maturity Model (CMM) defined in *Capability Maturity Model for Software, Version 1.1* [Paulk 93a]. It is compliant with the CMM Appraisal Framework (CAF) [Masters 95]. This document is an update to SCE Version 2.0 [CBA Project 94].

http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.003.html

Special Reports

CMU/SEI-96-SR-011
ADA308240

Directory of Industry and University Collaborations with a Focus on Software Engineering Education

Beckman, K. (Computer Data Systems, Inc.)

This directory describes collaborative efforts to promote software engineering education among businesses and universities in the United States and Canada. The groups vary in their maturity and the services they provide. The reader can use this directory to find collaborations that match their needs and are located within their geographic area.

http://www.sei.cmu.edu/publications/documents/96.reports/96.sr.011.html

CMU/SEI-96-SR-006
ADA314007

SEI Strategic Plan: 1997-2001

This document presents the strategic plan of the Software Engineering Institute (SEI) for the next five years (1997-2001). The SEI technical program is organized into three broad areas: technical engineering practices, enhanced software management capabilities, and transition readiness. Because technical engineering practices potentially cover a very wide set of issues, we intend to use information survivability as a unifying application problem for this aspect of our work. This document was written in early 1996 and delivered to our sponsor (the Defense Advanced Research Projects Agency [DARPA]) as a contract deliverable in July 1996. As such, it was a draft plan; its execution depends primarily on approved resource allocations.The planning starts long before the Congress completes its budget authorization and appropriation. Historically, circumstances such as changing customer needs and shifting resource allocations have made it necessary to change our plans.

http://www.sei.cmu.edu/publications/documents/96.reports/96.sr.006.html

CMU/SEI-96-SR-004
ADA319072

Software Acquisition Capability Maturity Model[SM] Pilot Appraisal Report, Version 1.0

Ferguson, J.R.; Jones, L.G.; Philpot, J.

This document summarizes five pilot appraisals performed from the third quarter of 1995 through the first quarter of 1996 using the Software Acquisition Capability Maturity Model (SA-CMM). The pilot appraisals used Version 0.01 of the SA-CMM, published in June 1995; Version 0.02 of the SA-CMM, published in February 1996; and the CMM-based appraisal methodology for internal process improvement (CBA IPI), tailored for use with the SA-CMM.

http://www.sei.cmu.edu/publications/documents/96.reports/96.sr.004.html

CMU/SEI-96-SR-003
ADA324232

CMM[SM] Version 1.1 Measurement Map

Park, R.E.

This report identifies and tabulates all references to software measures and measurement activities in Version 1.1 of the Capability Maturity Model[SM] for Software (CMM[SM]). Each reference is listed in a structured format, and the results are sorted in a way that is designed to help organizations plan the evolution of their measurement activities across the key process areas of the CMM. Where the guidance in the CMM is unclear or incomplete, opportunities for improving the CMM are noted and explained.

CMU/SEI-96-SR-001
ADA311456

Domain Analysis Workshop Report for the Automated Prompt and Response System Domain

Krut Jr., R.; Zalman, N.

This report captures the results of the domain analysis tutorial and workshop at Research Triangle Park for Bell Northern Research/Northern Telecom (BNR/NT). Included in this report are brief descriptions of the components of the domain analysis methodology employed and the products developed during the workshop. The information captured within this report will serve as a supplement to the Feature-Oriented Domain Analysis (FODA) tutorial and workshop for the ultimate pilot study domain. The importance of this report is that it provides the future workshop participants with an understanding of the types of products created at a workshop and gives examples of FODA products in a domain familiar to the participants.

http://www.sei.cmu.edu/publications/documents/96.reports/96.sr.001.html

Handbooks

CMU/SEI-96-HB-004
ADA293345

Description of the Systems Engineering Capability Maturity Model Appraisal Method Version 1.1, A

Kuhn, D.; Wells, C.; Armitage, J.; Cusick, K.; Garcia, S.; Hanna, M.; Malpass, P.; Pierson, H.

The purpose of this document is to summarize the major elements of the Systems Engineering Capability Maturity Model (SE-CMM) appraisal method (SAM). SAM is a method for using the SE-CMM to benchmark, or otherwise appraise, the process capability of an organization's or enterprise's systems engineering function. The SE-CMM itself is described in CMU/SEI-96-HB-004 [SECMM]. This document describes each step of an SE-CMM appraisal and provides guidance for the preparation and conduct of an appraisal. It also contains background and context information about the appraisal method.

http://www.sei.cmu.edu/publications/documents/96.reports/96.hb.001.html

CMU/SEI-96-HB-002
ADA313946

Goal-Driven Software Measurement—A Guidebook

Park, R.E.

The materials in this guidebook are designed to help you identify, select, define, and implement software measures to support your business goals. The measures that result are traceable back to your business goals, so that data collection efforts are better able to stay focused on their intended objectives.

http://www.sei.cmu.edu/publications/documents/96.reports/96.hb.002.html

CMU/SEI-96-HB-001
ADA305472

*IDEAL*SM *A User's Guide for Software Process Improvement*

McFeeley, B.

This document describes a software process improvement (SPI) program model, IDEAL, which can be used to guide development of a long-range, integrated plan for initiating and managing an SPI program. The purpose of this document is to provide process improvement managers with a generic description of a sequence of recommended steps for SPI.

http://www.sei.cmu.edu/publications/documents/96.reports/96.hb.001.html

1995 Reports

Technical Reports

CMU/SEI-95-TR-021
ADA307888

Quality Attributes

Barbacci, M.; Klein, M.; Longstaff, T.; Weinstock, C.

Computer systems are used in many critical applications where a failure can have serious consequences (loss of lives or property). Developing systematic ways to relate the software quality attributes of a system to the system's architecture provides a sound basis for making objective decisions about design trade-offs and enables engineers to make reasonably accurate predictions about a system's attributes that are free from bias and hidden assumptions. The ultimate goal is the ability to quantitatively evaluate and trade off multiple software quality attributes to arrive at a better overall system. The purpose of this report is to take a small step in the direction of developing a unifying approach for reasoning about multiple software quality attributes. In this report, we define software quality, introduce a generic taxonomy of attributes, discuss the connections between the attributes, and discuss future work leading to an attribute-based methodology for evaluating software architectures.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.021.html

CMU/SEI-95-TR-020
ADA305400

State of the Practice Report: Problems in the Practice of Performance Engineering

Klein, M.

As systems have performance requirements, sometimes dominant and explicit, and other times subordinate and implicit. Despite the pervasiveness and importance of performance requirements, performance problems persist. To help us understand why6, we sponsored a workshop in performance engineering and conducted some structured interviews with software contractors. This report summarizes our observations.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.020.html

CMU/SEI-95-TR-019
ADA309159

Evolutionary Perspective of Software Engineering Research Through Co-Word Analysis, An

Coulter, N.; Monarch, I.; Konda, S.; Carr, M.

This study applies various tools, techniques, and methods that the Software Engineering Institute is evaluating for analyzing information being produced at a very rapid rate in the discipline—both in practice and in research. The focus here is on mapping the evolution of the research literature as a means to characterize software engineering and distinguish it from other disciplines. Software engineering is a term often used to describe programming-in-the-large activities. Yet, any precise empirical characterization of its conceptual contours and their evolution is lacking. In this study, a large number of publications from 1982-1994 are analyzed to determine themes and trends in software engineering.

The method used to analyze the publications was co-word analysis. This methodology identifies associations among publication descriptors (indexing terms) from the Computing Classification System and produces networks of terms that reveal patterns of associations. The results suggest that certain research themes in software engineering remain constant, but with changing thrusts. Other themes mature and then diminish as major research topics, while still others seem transient or immature. Certain themes are emerging as predominate for the most recent time period covered (1991-1994): object-oriented methods and user interfaces are identifiable as central themes.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.019.html

CMU/SEI-95-TR-018
ADA310914

Fingertip Access to Software Engineering Information and Learning: SAIL on the Informedia DVLS

Hallman, H.

Practicing software engineers have difficulty accessing state-of-the-practice technologies in a timely manner. As a result, software engineers frequently re-invent the technology. Fingertip information (project, state-of-the-practice, state-of-the-art, domain specific, etc.) should be provided so that software engineers perform their profession more effectively. The System for Access to Information and Learning (SAIL) approach on the Informedia™ Digital Video Library System (DVLS) demonstrates the feasibility of providing fingertip access to information and learning materials, using requirements elicitation as the technology base. Also, Informedia DVLS and the World Wide Web can supplement each other; by using the Web to gain access to certain areas of software engineering, we can begin to assemble libraries of technical information. Informedia DVLS can provide timely access to such information and can also be used to house current project materials.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.018.html

CMU/SEI-95-TR-017
ADA303319

Raytheon Electronic Systems Experience in Software Process Improvement

Haley, T.; Ireland, B.; Wojtaszek, E.; Nash, D.; Dion, R.

The software engineering process group (SEPG) of Raytheon Electronic Systems (RES) is responsible for defining and implementing the Software Engineering Initiative, which outlines the policies, practices, and procedures to be followed in developing complex software for large-scale commercial and defense projects. To accomplish these objectives, the SEPG has had to develop the organizational structure and techniques to meet the growing challenges of developing, maintaining, and improving its software engineering process in significant and measurable ways, including quantifying return on investment (ROI) and increasing the quality of the deliverable product.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.017.html

CMU/SEI-95-TR-016
ADA309157

Collaboration in Implementing Team Risk Management, A

Gluch, D.; Dorofee, A.; Hubbard, E.; Travalent, J.

This report presents results of a collaborative development effort to transition the Software Engineering Institute (SEI) team risk management process into practice. The collaboration involved a DoD program office (customer), a commercial contractor (supplier), and the SEI in an effort that included development, test, and refinement of the team risk management approach. The focus of the report is on the results of the collaboration between the contractor organization and the SEI.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.016.html

CMU/SEI-95-TR-015
ADA311066

Unified Information Security (INFOSEC) Architecture (UIA) Gadfly Project, The

Maymir-Ducharme, F.; Clements, P.C.; Wallnau, K.C.; Krut Jr., R.

This report captures the development, lessons learned, and future recommendations from a collaborative research and development activity between the Air Force sponsored Comprehensive Approach to Reusable Defense Software (CARDS) Program, the Department of Defense (DoD), and the Software Engineering Institute (SEI). This activity explored innovative but practical techniques for formalizing and applying (i.e., reusing) models of information security (INFOSEC) concepts to the development of information systems.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.015.html

CMU/SEI-95-TR-014
ADA302320

Experiment in Software Development Risk Information Analysis, An

Monarch, I.

The following report summarizes the results of an experiment that uses terminological structures derived from the application of knowledge summarization, analysis, and visualization (K-SAV) technology to textual data from the Software Engineering Risk Repository (SERR) resident at the Software Engineering Institute. This study evaluates the use of several tools including shared word clustering and a co-word analysis software program, leximappe. The experiment seeks to determine whether an application of co-word analysis to baseline risk assessment data would enable a reduction of the information load while simultaneously providing a succinct but encompassing picture of the risk information within the program. This study is based upon a somewhat limited data set. Nevertheless, the results of this investigation are encouraging and suggest that there may be value and potential for the effective use of co-word analysis and K-SAV technology more generally in risk management. Additional investigations are underway to confirm, alter, or challenge the results.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.014.html

CMU/SEI-95-TR-012
ADA315789

Software Capability Evaluation Version 3.0 Implementation Guide for Supplier Selection

Barbour, R.

This report describes implementation guidance for Version 3.0 of the Software Capability Evaluation (SCE) method. This version of the Implementation Guide is updated to reflect the new method and provides specific guidance for selecting software product and services suppliers in an acquisition application (government or commercial) and provides suggested language and examples of usage.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.012.html

CMU/SEI-95-TR-011
ADA304100

Distributed System Design Using Generalized Rate Monotonic Theory

Sha, L.; Sathaye, S.

Rate monotonic theory and its generalizations have been adopted by national high technology projects such as the space station and have recently been supported by major open standards such as the IEEE Futurebus+ and POSIX.4. In this paper, we describe the use of generalized rate monotonic scheduling theory for the design and analysis of a distributed real-time system. We review the theory, examine the architectural requirements for the use of the theory, and finally provide an application example.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.011.html

CMU/SEI-95-TR-010
ADA304091

Training Guidelines: Purchasing Training for a Software Organization

Carpenter, M.

This set of training guidelines focuses on many of the issues surrounding the purchasing of software engineering training. It includes general guidelines for creating partnerships between training vendors and training purchasers to meet real training needs and a template for use by subject matter experts in elaborating training requirements.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.010.html

CMU/SEI-95-TR-009
ADA302225

After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success

Goldenson, D.; Herbsleb, J.

Very little published evidence exists about the impact of the Capability Maturity Model[SM] (CMM[SM]) or CMM-based appraisals on subsequent software process improvement and organizational performance. A few credible case studies do exist, but it is uncertain how widely their results apply. We present evidence here from a much broader cross section of software organizations. Our results suggest that process maturity does indeed payoff in better product quality, ability to meet schedule commitments, and other indicators of organizational performance. The vast majority of survey respondents also report that their appraisals proved to be highly accurate and useful in guiding their subsequent process improvement efforts. Not all organizations have been equally successful, however, and improvement often takes longer and costs more than expected. We identify several factors, most of them under management control, that distinguish more successful from less successful organizations.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.009.html

CMU/SEI-95-TR-008
ADA300121

Moving On Up: Data and Experience Doing CMM-Based Process Improvement

Hayes, W.; Zubrow, D.

An analysis of Software Process Assessment results from 48 organizations undertaking 2 or more assessments is presented in this report. The analysis focuses on the time required to increase process maturity, as well as the most prevalent process issues faced the 48 organizations. Results of the analysis are used to provide guidance to organizations embarking on a software process improvement effort.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.008.html

CMU/SEI-95-TR-007
ADA300120

Training Guidelines: Creating a Training Plan for a Software Organization

Carpenter, M.

The key process areas at Level 3 address both project and organizational issues, as the organization establishes an infrastructure that institutionalizes effective software engineering and management processes across all projects. The purpose of the Training Program is to develop the skills and knowledge of individuals so they can perform their roles effectively and efficiently. Training is an organizational responsibility, but the software projects are responsible for identifying their needed skills and providing the necessary training when the project's needs are unique.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.007.html

CMU/SEI-95-TR-005
ADA301169

Software Architecture for Dependable and Evolvable Industrial Computing Systems, A

Sha, L.

The downtime of a large industrial operation is often prohibitively expensive and a failure of a mission critical system could have disastrous consequences. Lacking an effective approach to mitigate the risks in system upgrades or to introduce third party supplied open system components, many industrial systems and defense systems are forced to keep outdated computing hardware and software. A paradigm shift is needed, from a focus on enabling technologies for completely new installations to one which is designed to mitigate the risk and cost of bringing new technology into functioning systems. Innovative technology is needed to support the task of *technology insertion*. Quickly and reliably turning unparalleled American innovations into industrial competitiveness and defense technological superiority is of strategic importance.
The Simplex architecture has been developed to support safe and reliable online upgrade of hardware and software components in spite of errors in the new modules. This paper gives a brief overview of the underlying technologies.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.005.html

CMU/SEI-95-TR-004
ADA300233

Report on Distance Learning Technologies

Capell, P.

This report provides a wide view of the costs, risks, and benefits associated with instructional technology alternatives. The number and variety of possible paths to learning through this technology have increased markedly in recent years with the advent of interactive multimedia, satellite communications, and the Internet. More than ever before, learning technologies have created new educational possibilities for people and organizations. In this era of increasing financial stringency, we are obligated to examine these new possibilities in light of their obvious advantages: replication of high-quality instruction, lower overall costs, increased quality in educational outcomes, and the ability to provide these benefits over long distances. This report will show that with today's computer-based instructional technology, the question is no longer whether to use the technology, but rather how to use it.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.004.html

CMU/SEI-95-TR-003
ADA296804

Subject Matter of Process Improvement: A Topic and Reference Source for Software Engineering Educators and Trainers, The

Ibrahim, R.L.; Hirmanpour, I.

This report provides a high-level topical overview of what can be taught or learned about process improvement. The subject matter is presented within a general framework of six major topic areas, which are described and divided into annotated subtopics. The relationships and application of the subject areas are explained in the context of process improvement activities. Topic areas range from process and process improvement concepts to tools, techniques, teamwork, and interpersonal skills.

The purpose of this report is to assist software engineering educators and trainers in selecting topics for curricula or training programs. It may also be used to guide self-study in this area. Pointers to detailed sources of information are given, but no in-depth information is otherwise provided for the topic areas. Consequently, this report is not suitable for use by itself as a means of learning the details of how to do process improvement.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.003.html

CMU/SEI-95-TR-002
ADA294737

Object-Oriented Software Measures

Archer, C.,; Stinson, M.

This paper provides an overview of the merging of a paradigm and a process, the object-oriented paradigm and the software product measurement process. A taxonomy of object-oriented software measures is created, and existing object-oriented software measures are enumerated, evaluated, and placed in taxa. This report includes an extensive bibliography of the current object-oriented measures that apply to the design and implementation phases of a software project. Examples of computation of representative measures are included.

http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.002.html

CMU/SEI-95-TR-001
ADA293300

CMM Appraisal Framework, Version 1.0

Masters, S.; Bothwell, C.

This technical report describes version 1.0 of the CMM Appraisal Framework (CAF). This framework describes the common requirements used by the CMM-Based Appraisal (CBA) project in developing appraisal methods based on the Capability Maturity Model CMM) for Software, Version 1.1. The RCAF provides a framework for rating the process maturity of an organization against the CMM. The CAF includes a generic appraisal architecture for CMM-based appraisal methods and defines the requirements for developing CAF compliant appraisal methods.

http://www.sei.cmu.edu/publications/documents/95.reports/95-tr-001/95-tr-001-abstract.html

Special Reports

CMU/SEI-95-SR-018
ADA302319

Information Technology—Programming Language—The SQL ADA Module Description Language (SAMeDL)—ISO/IEC 12227

ISO/IEC 12227

This International Standard specifies the syntax and semantics of a database programming language, the SQL ADA Module Description Language, SAMeDL. Texts written in the SAMeDL describe database interactions which are to be performed by database management systems (DBMS) implementing Database Language SQL. The interactions so described and so implemented are to be performed on behalf of application programs written in Programming Language ADA.

http://www.sei.cmu.edu/publications/documents/95.reports/95.sr.018.html

CMU/SEI-95-SR-011
ADA300779

Directory of Industry and University Collaborations with a Focus on Software Engineering Education

Carpenter, M.

This directory describes collaborative efforts among businesses and universities in the United States and Canada to promote software engineering education. The groups vary in their maturity and the services provided. This directory points organizations toward collaborations within their geographic area.

http://www.sei.cmu.edu/publications/documents/95.reports/95.sr.011.html

CMU/SEI-95-SR-007
ADA299527

Proceedings of the SEI/MCC Symposium on the Use of COTS in Systems Integration

Brown, A.; Carney, D.; McFalls, M. (editors)

The SEI/MCC Symposium on the Use of COTS (commercial off-the-shelf) in Systems Integration took place at the Software Engineering Institute (SEI) in Pittsburgh, Pennsylvania, on January 10-11, 1995. The symposium focused on two key points: the Department of Defense need for integrated systems, and its increasing reliance on acquiring software through commercial sources. The interrelationships between these points formed the conceptual basis for this symposium. These proceedings provide a record of the presentations and some of the main highlights from the discussions. The main body of this report is a set of notes from each of the panels.

http://www.sei.cmu.edu/publications/documents/95.reports/95.sr.007.html

CMU/SEI-95-SR-005
ADA293299

Checklists and Criteria for Evaluating the Cost and Schedule Estimating Capabilities of Software Organizations

Park, R.

This report provides criteria and checklists for evaluating the capability of an organization's software estimating process and the infrastructure that supports it. It also supplies guidelines for good estimating practice. The checklists and guidelines can be used to elicit information for process assessments and to motivate and guide organizations in process improvement efforts.

http://www.sei.cmu.edu/publications/documents/95.reports/95.sr.005.html

CMU/SEI-95-SR-004
ADA293298

Manager's Checklist for Validating Software Cost and Schedule Estimates, A

Park, R.

This report provides a checklist of questions to ask and evidence to look for when assessing the credibility of a software cost and schedule estimate. The checklist can be used either to review individual estimates or to motivate and guide organizations toward improving their software estimating processes and practices.

http://www.sei.cmu.edu/publications/documents/95.reports/95.sr.004.html

Maturity Models

CMU/SEI-95-MM-003
ADA303318

Systems Engineering Capability Maturity Model, Version 1.1, A

The Systems Engineering Capability Maturity Model^{SM} (SE-CMM^{SM}) describes the essential elements of an organization's systems engineering process that must exist to ensure good systems engineering. It does not specify a particular process or sequence. In addition, the SE-CMM provides a reference for comparing actual systems engineering practices against these essential systems.

This document provides an overall description of the principles and architecture upon which the SE-CMM is based, an overview of the model, the practices included in the model, and a description of the attributes of the model. It also includes the requirements used to develop the model.

http://www.sei.cmu.edu/publications/documents/95.reports/95.mm.003.html

CMU/SEI-95-MM-002
ADA300822

People Capability Maturity Model^{SM}

Curtis, B.; Hefley, W.; Miller, S.

The People Capability Maturity Model^{SM} (P-CMM^{SM}) adapts the maturity framework of the Capability Maturity Model^{SM} for Software (CMM^{SM}) [Paulk 95] to managing and developing an organization's workforce. The motivation for the P-CMM is to radically improve the ability of software organizations to attract, develop, motivate, organize, and retain the talent needed to continuously improve software development capability. The P-CMM is designed to allow software organizations to integrate workforce improvement with software process improvement programs guided by the CMM. The P-CMM can also be used by any kind of organization as a guide for improving its people-related and workforce practices.

Based on the best current practices in fields such as human resources and organizational development, the P-CMM provides organizations with guidance on how to gain control of their processes for managing and developing their workforce. The P-CMM helps organizations to characterize the maturity of their workforce practices, guide a program of continuous workforce development, set priorities for immediate actions, integrate workforce development with process improvement, and establish a culture of software engineering excellence. It describes an evolutionary improvement path from ad hoc, inconsistently performed practices, to a mature, disciplined development of the knowledge, skills, and motivation of the workforce, just as the CMM describes an evolutionary improvement path for the software processes within an organization.

The P-CMM consists of five maturity levels that lay successive foundations for continuously improving talent, developing effective teams, and successfully managing the people assets of the organization. Each maturity level is a well-defined evolutionary plateau that institutionalizes a level of capability for developing the talent within the organization. The key process areas at Level 2 focus on instilling basic discipline into workforce activities. They are Work Environment, Communication, Staffing, Performance Management, Training, and Compensation. The key process areas at Level 3 address issues surrounding the identification of the organization's primary competencies and aligning its people management activities with them. They are Knowledge and Skills Analysis, Workforce Planning, Competency Development, Career Development, Competency-Based Practices, and Participatory Culture. The key process areas at Level 4 focus on quantitatively managing organizational growth in people management capabilities and in establishing competency-based teams. They are Mentoring, Team Building, Team-Based Practices, Organizational Competency Management, and Organizational Performance Alignment. The key process areas at Level 5 cover the issues that address continuous improvement of methods for developing competency, at both the organizational and the individual level. They are Personal Competency Development, Coaching, and Continuous Workforce Innovation.

This document describes the P-CMM, the key practices that correspond to each maturity level of the P-CMM, and information on how to apply the P-CMM in guiding organizational improvement. It contains an elaboration of what is meant by workforce capability (i.e., maturity) at each maturity level, and describes how the P-CMM can be applied by an organization in two primary ways: as a standard for assessing workforce practices, and as a guide in planning and implementing improvement activities.

http://www.sei.cmu.edu/publications/documents/95.reports/95.mm.003.html

CMU/SEI-95-MM-001
ADA301167S

Overview of the People Capability Maturity Model, Version 1.1

Curtis, B.; Hefley, W.; Miller, S.

The People Capability Maturity Model^{SM} (P-CMM^{SM}) adapts the maturity framework of the Capability Maturity Model^{SM} for Software (CMM^{SM}) [Paulk 95], to managing and developing an organization's workforce. The motivation for the P-CMM is to radically improve the ability of software organizations to attract, develop, motivate, organize, and retain the talent needed to continuously improve software development capability. The P-CMM is designed to allow software organizations to integrate workforce improvement with software process improvement programs guided by the CMM. The P-CMM can also be used by any kind of organization as a guide for improving their people-related and workforce practices.

Based on the best current practices in the fields such as human resources and organizational development, the P-CMM provides organizations with guidance on how to gain control of their processes for managing and developing their workforce. The P-CMM helps organizations to characterize the maturity of their workforce practices, guide a program of continuous workforce development, set priorities for immediate actions, integrate workforce development with process improvement, and establish a culture of software engineering excellence. It describes an evolutionary improvement path from ad hoc, inconsistently performed practices, to a mature, disciplined development of the knowledge, skills, and motivation of the workforce, just as the CMM describes an evolutionary improvement path for the software processes within an organization.

The P-CMM consists of five maturity levels that lay successive foundations for continuously improving talent, developing effective teams, and successfully managing the people assets of the organization. Each maturity level is a well-defined evolutionary plateau that institutionalizes a level of capability for developing the talent within the organization.

This document provides an overview of the P-CMM. The P-CMM and its key practices are fully described in the People Capability Maturity Model^{SM} (CMU/SEI-95-MM-002, September 1995).

http://www.sei.cmu.edu/publications/documents/95.reports/95.mm.003.html

Curriculum Modules and Support Materials

SEI-CM-28

Measuring Object-Oriented Software Products

Archer, C. (Winthrop University)

June 1995

This module provides an overview of the merging of a paradigm and a process, the object-oriented paradigm and the software measurement process. The concept of a measure and the process of measurement are discussed briefly, followed by a presentation of the issues raised by object-oriented software development.

http://www.sei.cmu.edu/publications/documents/cms/cm.028.html

1994 Reports

Technical Reports

CMU/SEI-94-TR-026
ADA303272

Relationships Between the Systems Engineering Capability Maturity Model and Other Products, Version 1.0

The SE-CMM is a document that describes characteristics, both domain and process-management focused, of systems engineering processes that contribute to successful product development. From the beginning of the effort, users of the SE-CMM have requested information on how SE-CMM practices relate to other products. This document is an initial effort at identifying and characterizing these relationships.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.026.html

CMU/SEI-94-TR-024
ADA302689

Process Tailoring and the Software Capability Maturity Model

Ginsberg, M.; Quinn, L.

The Software Capability Maturity Model^{SM} (SW-CMM^{SM}) is serving as the foundation for a major portion of the process improvement being undertaken in the software industry. It is composed of two volumes: the Capability Maturity Model for Software and the Key Practices of the Capability Maturity Model. The key practices of the SW-CMM are expressed in terms that reflect normal practices of organizations that work on large, government contracts. There is, however, a significant population of software-producing and acquiring organizations, operating in different environments, for which the key practices require significant interpretation and/or tailoring, prior to application. This report presents a tailoring framework that identifies process artifacts, tailoring processes, and their relationships to project artifacts, and explores the nature of various kinds of tailoring used in the definition and development of software process descriptions. Techniques appropriate to each type of tailoring are then discussed. The general approach utilizes and builds upon the Software Process Framework, whose purpose is to provide guidance for designing, analyzing, and reviewing software processes for consistency with the SW-CMM.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.024.html

CMU/SEI-94-TR-023
ADA292215

Characteristics of Higher-Level Languages for Software Architecture

Shaw, M.; Garlan, D.

As the size and complexity of software systems increases, the design and specification of overall system structure—or software architecture—emerges as a central concern. Architectural issues include the gross organization of the system, protocols for communication and data access, assignment of functionality to design elements, and selection among design alternatives.

Currently, system designers have at their disposal two primary ways of defining software architecture: they can use the modularization facilities of existing programming languages and module inter connection languages; or they can describe their designs using informal diagrams and idiomatic phrases (such as “client-server organization”).

In this paper, we explain why neither alternative is adequate. We consider the nature of architectural description as it is performed informally by systems designers. Then we show that regularities in these descriptions can form the basis for architectural description languages. Next, we identify specific properties that such languages should have. Finally, we illustrate how current notations fail to satisfy those properties.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.023.html

CMU/SEI-94-TR-022
ADA289912

Software Process Improvement in the NASA Software Engineering Laboratory

McGarry, F.; Pajerski, R.; Page, G.; Waligora, S.; Basili, V.; Zelkowitz, M.

The Software Engineering Laboratory (SEL) was established in 1976 for the purpose of studying and measuring software processes with the intent of identifying improvements that could be applied to the production of ground support software within the Flight Dynamics Division (FDD) at the National Aeronautics and Space Administration (NASA)/Goddard Space Flight Center (GSFC). The SEL has three member organizations: NASA/GSFC, the University of Maryland, and Computer Sciences Corporation. The concept of process improvement within the SEL focuses on the continual understanding of both process and product as well as goal-driven experimentation and analysis of process change within a production environment.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.022.html

CMU/SEI-94-TR-021
ADA288963

Introduction to Software Architecture, An

Garlan, D.; Shaw, M.

As the size of software systems increases, the algorithms and data structures of the computation no longer constitute the major design problems. When systems are constructed from many components, the organization of the overall system—the software architecture—presents a new set of design problems. This level of design has been addressed in a number of ways, including informal diagrams and descriptive terms, module interconnection languages, templates, and frameworks for systems that serve the needs of specific domains, and formal models of component integration mechanisms. In this paper, we provide an introduction to the emerging field of software architecture. We begin by considering a number of common architectural styles upon which many systems are currently based, and how different styles can be combined in a single design. Then, we present six case studies to illustrate how architectural representations can improve our understanding of complex software systems. Finally, we survey some of the outstanding problems in the field, and consider a few of the promising research directions.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.021.html

CMU/SEI-94-TR-020
ADA293020

Experience with a Course on Architectures for Software Systems Part II: Educational Materials

Shaw, M.; Garlan, D.; Gaimes, J.

This report contains the materials used by the instructors to teach the course CS 15-775: Architectures for Software Systems in the Spring of 1994 in the School of Computer Science at Carnegie Mellon University. The materials include the lecture slides, questions (with answers) on readings, and homework assignments (with sample solutions.)

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.020.html

CMU/SEI-94-TR-018
ADA288708

Spinning a Web: Publishing the SEI Software Configuration Management Research on the World Wide Web

Huff, C.

Configuration Management research has been performed by members of the CASE Environments Project over the course of the past five years. This report describes the contents of the configuration management research materials that have been published on the SEI World Wide Web (WWW) Server. Primary Web Structures and methods for accessing information on the Web are described. A summary of the problems and challenges encountered and the Web publishing techniques employed in this process are discussed.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.018.html

CMU/SEI-94-TR-017
ADA289928

Replacing the Message Service Component in an Integration Framework

Zarrella, P.; Brown, A.

In an on-going set of commercial off-the-shelf (COTS) tool integration experiments being conducted by the CASE Environments Project, we have integrated a set of CASE tools using a combination of data integration mechanisms (PCTE Object Management System (QMS) and UNIX file system) and control integration mechanisms (Broadcast Message Server (BMS) of HP SoftBench). One of the key issues addressed in our work is the extent to which the integration of CASE tools can be independent of particular integration framework technology products.

This report describes a task to examine interoperability aspects of the control integration component of the integration framework. The major conclusion from our work is that it is possible to integrate CASE tools using a message-passing approach that is independent of the integration framework product used. This report describes the activities an organization must undertake to integrate CASE tools in order to ensure this interoperation of message-passing integration products. The report also includes a set of lessons learned concerning the experiments we carried out.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.017.html

CMU/SEI-94-TR-016
ADA296801

DoD Software Measurement Pilot: Applying the SEI Core Measures, A

Rozum, J.; Florac, W.

A pilot effort was initiated by the U.S. Department of Defense (DoD) and led by the Defense Information Systems Agency (DISA) to assess the issues and effort involved in implementing a software measurement program across multiple sites and projects. The pilot was conducted involving multiple DoD organizations and projects from varying application domains. The objectives were to assess the ability of an organization to assess and use the core measures, determine the appropriate analysis and reporting of the measurement information at an organization and corporate level, determine the applicability of common definitions of the SEI core software measures, evaluate the effectiveness of the SEI core measure checklists for development and maintenance, and develop metrics program guidelines for DoD implementation. This technical report discusses the observations and lessons learned from the pilot effort and makes recommendations regarding software measurement implementation across a large organization.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.016.html

CMU/SEI-94-TR-015
ADA311066

Beyond Objects: A Software Design Paradigm Based on Process Control

Shaw, M.

A standard demonstration problem in object-oriented programming is the design of an automobile cruise control. This design exercise demonstrates object-oriented techniques well, but it does not ask whether the object-oriented paradigm is the best one for the task. Here, we examine the alternative view that cruise control is essentially a control problem. We present a new software organization paradigm motivated by process control loops. The control view leads us to an architecture that is dominated by analysis of a classical feedback loop rather than by the identification of discrete stateful components to treat as objects. The change in architectural model calls attention to important questions about the cruise control task that aren't addressed in an object-oriented design.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.015.html

CMU/SEI-94-TR-014
ADA284922

Construct for Describing Software Development Risks, A

Gluch, D.

This report establishes a representation of software risk wherein the risks associated with software-dependent development programs are defined as distinct, manageable risk entities. The risk entities and their descriptive statements of risk are based upon a Condition-Transition-Consequence (CTC) construct. The CTC construct arises out of a systems representation, where time and value are identified as fundamental to the concept of risk. The CTC construct is also shown to provide a common representation for both program risks and program tasks and

to fit into a heuristic framework for identifying risks within software-dependent development programs. Examples of risks are used to demonstrate that the approach facilitates the management of risk as an integral part of routine program management.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.014.html

CMU/SEI-94-TR-013
ADA283848

Benefits of CMM-Based Software Process Improvement: Initial Results

Herbsleb, J.; Carleton, A.; Rozum, J.; Siegel, J.; Zubrow, D.

Data from 13 organizations were collected and analyzed to obtain information on the results of CMM-based software process improvement efforts. We report the cost and business value of improvement efforts, as well as the yearly improvement in productivity, early defect direction, time to market, and post-release defect reports. Improvement efforts and results in five organizations are reported in more depth in case studies. In addition, technical issues that we confronted as we tried to measure the results of software process improvement are discussed. We end with conclusions about the results of SPI efforts.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.013.html

CMU/SEI-94-TR-012
ADA290697

Comparison of ISO 9001 and the Capability Maturity Model for Software, A

Paulk, M.

The Capability Maturity Model for Software (CMM), developed by the Software Engineering Institute, and the ISO 9000 series of standards, developed by the International Standards Organization, share a common concern with quality and process management. The two are driven by similar concerns and intuitively correlated. The purpose of this report is to contrast the CMM and ISO 9001, showing both their differences and their similarities. The results of the analysis indicate that, although an ISO 9001-compliant organization would not necessarily satisfy all of the level 2 key process areas, it would satisfy most of the level 2 goals and many level 3 goals. Because there are practices in the CMM that are not addressed in ISO 9000, it is possible for a level 1 organization to receive 9001 registration; similarly, there are areas addressed by ISO 9001 that are not addressed in the CMM. A level 3 organization would have little difficulty in obtaining ISO 9001 certification, and a level 2 organization would have significant advantages in obtaining certification.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.012.html

CMU/SEI-94-TR-011
ADA280940

Progress Report on Undergraduate Software Engineering Education, A

Ford, G.

The current status of undergraduate software engineering education in United States universities is summarized, including descriptions of programs at eleven schools. Possible scenarios for the further evolution of undergraduate software engineering programs are described, based on observations of the evolution of computer science and computer engineering programs. Recent and ongoing activities of the Computer Society of the Institute of Electrical and Electronics Engineers (IEEE) and the Association for Computing Machinery (ACM) regarding the establishment of the profession of software engineering are described, including the expected implications for undergraduate software engineering education.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.011.html

CMU/SEI-94-TR-010
ADA283827

Toward Deriving Software Architectures From Quality Attributes

Kazman, R.; Bass, L.

A method for deriving software architectures from a consideration of the non-functional qualities of the system is presented. The method is based on identifying a set of six “unit operations” and using those operations to partition the functionality of the system. These unit operations were derived from the literature and from expert practice.

The relationship between the unit operations and a set of eight non-functional qualities is explored. Evidence is provided for the validity of the method by using it to derive six well-known architectures from the areas of user interface software and compiler constructions.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.010.html

CMU/SEI-94-TR-009
ADA286093

Artificial Intelligence (AI) and ADA: Integrating AI with Mainstream Software Engineering

Diaz-Herrera, J.

In this report we discuss in detail pragmatic problems posed by the integration of AI with conventional software engineering, and within the framework of current ADA technology. A major objective of this work has been to begin to bridge the gap between the ADA and AI software cultures. The report summarizes survey results from the Association for Computing Machinery (ACM) Special Interest Group for ADA (SIGADA) AI Working Group (AIWG), highlighting lessons learned and sample applications. An interesting observation is that a large percentage of AI code is procedural by nature and that better productivity rates are achieved by using ADA; also, efficiency is much better when compared to traditional AI languages. Although we show favorable results on the use of ADA technology for the implementation of AI software, a total integration remains difficult at the conceptual level. There are some impediments to a completely satisfactory solution, but only a few restrictions are more intrinsically related to the current ADA standard (ADA83); these, however, are being dealt with in the next revision known as ADA9X.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.009.html

CMU/SEI-94-TR-008
ADA283747

Mapping a Domain Model and Architecture to a Generic Design

Peterson, A.; Stanley, Jr, J.

In contrast to the number of reports on domain analysis, little work has been done in describing the utilization of domain analysis results in the development of generic designs for building applications in a domain. This report describes a process for mapping domain information in Feature-Oriented Domain Analysis (FODA) into a generic design for a domain. The design includes supporting code components that conform to the Object Connection Architecture (OCA), a model for structuring software systems. A process for the use of the design in implementing applications is included. The processes and products described herein augment the final phase of domain analysis (or engineering) described in the original FODA report. This report also documents the continuing work of applying FODA to the movement control domain. The design and ADA code examples for the domain used in the document are from prototype software, created in part to test the processes presented.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.008.html

CMU/SEI-94-TR-007
ADA280916

Practical Guide to the Technology and Adoption of Software Process Automation, A

Christie, A.

Process automation provides a means to integrate people in a software development organization with the development process and the tools supporting that development. For many reasons, this new technology has the potential to significantly improve software quality and software development productivity. As yet, however, there is little practical experience in its day-to-day use. The main goal of this report is thus to provide information for organizations that are considering its adoption. For these reasons, the report aims to identify how process automation relates to both process improvement and CASE tools, to review in some detail two of the major commercial process automation products, and to address relevant organizational adoption issues. It is hoped that the report will help bridge the gap between those whose focus is software process improvement and those whose focus is software technology.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.007.html

CMU/SEI-94-TR-006
ADA280943

Software Capability Evaluation Version 2.0 Method Description

This report describes Version 2.0 of the Software Capability Evaluation (SCE) Method, as taught at the Software Engineering Institute (SEI) from fourth quarter 2993. This version of the SCE Method is based on the Capability Maturity Model defined in *Capability Maturity Model for Software, Version 1.1* (CMU/SEI-91-TR-014). The document includes an overview of the SCE Method and its evolution, a detailed description of characteristics of the method and their implications for the use of the method. This document provides a new baseline for future evolution of the SCE Method.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.006.html

CMU/SEI-94-TR-005
ADA278635

Software Capability Evaluation (SCE) Version 2.0 Implementation Guide

Software Capability Evaluation (SCE) offers a means to evaluate an organization's software process capability—that is, how well an organization manages the process it used to create software. SCE provides a way to compare an offeror's software capability against a predefined standard. This document is an implementation guide: it is intended as a set of practical information which program managers can use to guide them through the process of using SCE in an acquisition.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.005.html

CMU/SEI-94-TR-004
ADA278596

Interim Profile Development and Trial of a Method to Rapidly Measure Software Engineering Maturity Status

Whitney, R.; Nawrocki, E.; Hayes, W.; Siegel, J.

Development of an Instant Profile (IP) method was driven by a business need to rapidly measure an organization's software engineering process maturity between organizational software process assessments (SPAs). This document provides information about the process used to develop the method and a description of the method to software engineering process group (SEPG) members and practitioners responsible for diagnosing software process maturity. This document also addresses the next steps in the further development and use of the instant profile method.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.004.html

CMU/SEI-94-TR-003
ADA28765

Exploring Hypermedia Information Services for Disseminating Software Engineering Information

Hefley, W.

This report describes the accomplishments of a pilot hypermedia information service embodying the conceptual definition of a pilot information base developed by the Software Engineering Institute (SEI) in support of the Advanced Research Projects Agency (ARPA) Software and Intelligent Systems Technology Office (SISTO). This pilot effort was conducted in support of the Technology Cost-Benefit Analysis tasks within TO&P 2-151, Software Engineering Technology Transition for Director, Defense Research and Engineering (DDR&E) and ARPA. This report also describes the intended uses and user populations of the proposed information base, design issues that influenced the structure and contents of the information base, a proposed information model consisting of information content and linkages, the pilot information base including the technology selected for the initial pilot effort and the pilot capability, lessons learned from the pilot effort, and future plans relating to the information base efforts. Key to these pilot efforts was the development of a set of proposed information structures for an information base on software engineering. These hypermedia-based information structures can be presented across the Internet and displayed on local workstations using client/server technologies, such as World Wide Web (WWW) and NCSA Mosaic (produced by the National Center for Supercomputing Applications).

This work to date has accomplished four goals. First, the objectives of the pilot effort have been met. An information base containing software engineering information that provides value to ARPA program managers has been demonstrated, and a work plan has been generated to expand from the pilot to an operational system. Second, the SEI has demonstrated a capability for effectively using the WWW, which is certain to be a critical part of the information highway for years to come. Third, techniques to enhance developers' productivity have been identified

and demonstrated. Preparation of online information can be aided by templates. Delivery of online information can be enhanced through study of actual users' navigational and usage patterns. Fourth, the SEI has established a "magnetic platform" as the facilities demonstrable at the SEI can be used as a starting point for developing new technology transition capabilities.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.003.html

CMU/SEI-94-TR-002
ADA281026

Procedure Calls Are the Assembly Language of Software Interconnection: Connectors Deserve First-Class Status

Shaw, M.

Software designers compose systems from components written in some programming language. They regularly describe systems using abstract patterns and sophisticated relations among components. However, the configuration tools at their disposal restrict them to composition mechanisms directly supported by the programming language. To remedy this lack of expressiveness, we must elevate the relations among components to first-class entities of the system, entitled to their own specifications and abstractions.

http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.002.html

Special Reports

CMU/SEI-94-SR-009
ADA286506

Software Acquisition: A Comparison of DoD and Commercial Practices

Ferguson, J.; DeRiso, M.

This paper will compare best commercial practice with the current Department of Defense (DoD) processes for acquiring software and to recommend some steps that can be taken to streamline DoD software acquisitions to minimize overall life-cycle costs.

http://www.sei.cmu.edu/publications/documents/94.reports/94.sr.009.html

CMU/SEI-94-SR-007
ADA285073

Maturity Questionnaire

Zubrow, D.; Hayes, W.; Siegel, J.; Goldenson, D.

This package contains a copy of the software process maturity questionnaire. It is intended for those interested in performing and learning about software process appraisals. This version differs in several important ways from its predecessor, *A Method for Assessing the Software Capability of Contractors* (CMU/SEI-87-TR-023). The most important difference is that this questionnaire is not an appraisal method itself; rather, it is one component that is uses in different appraisal methods.

http://www.sei.cmu.edu/publications/documents/94.reports/94.sr.007.html

CMU/SEI-94-SR-006
ADA283367

Second Dependable Software Technology Exchange

Weinstock, C.; Heimerdinger, W.

On March 24 and 25, 1994, the Open Attribute Engineering Project hosted the Second Dependable Software Technology Exchange. The exchange, sponsored by the Air Force Phillips Laboratories, the Office of Naval Research, and the Air Force Space and Missile Systems Center, brought together researchers and system developers, providing an opportunity for the researchers to learn the needs of the developers and for the developers to learn about techniques being investigated by the researchers. This report summarizes what transpired at the meeting.

http://www.sei.cmu.edu/publications/documents/94.reports/94.sr.006.html

CMU/SEI-94-SR-005
ADA283987

Team Risk Management: A New Model for Customer-Supplier Relationships

Higuera, R.; Dorofee, A.; Walker, J.; Williams, R.

This report will familiarize you with the concepts of Team Risk management by providing a description of the overall process that engages both the customer and supplier in a cooperative framework using explicit methods to manage project risks.

http://www.sei.cmu.edu/publications/documents/94.reports/94.sr.005.html

CMU/SEI-94-SR-003
ADA2800915

Software Cost and Schedule Estimating: A Process Improvement Initiative

Park, R.; Goethert, W.; Webb, J.

This report describes efforts that have been initiated by the Software Engineering Institute to improve the practice of software cost and schedule estimating. These efforts involve support and participation from both industry and government. They are motivated by the capability maturity model, which identified the key roles estimating and cost management play in establishing repeatable software processes. Products from the initiative will include templates, criteria, and guidelines for establishing defined estimating processes, training materials, and examples for teaching good estimating practice, and evaluations of the abilities of contemporary cost models to meet today's estimating needs.

http://www.sei.cmu.edu/publications/documents/94.reports/94.sr.003.html

CMU/SEI-94-SR-001
ADA285070

Introduction to Team Risk Management (Version 1.0), An

Higuera, R.; Gluch, D.; Dorofee, A.; Murphy, L.; Walker, A.; Williams, C.

(Version 1.0) Team Risk Management defines the organizational structure and operational activities for managing risks throughout all phases of the life-cycle of a software-dependent development program such that all individuals within the organizations, groups, departments, and agencies directly involved in the program are participating team members. Through the adoption of team risk management, the government and contractor are provided with processes, methods, and tools that enable both organizations, individually and jointly, to be increasingly anticipatory in decision-making processes. This report introduces the team risk management approach for managing risks within a software-dependent development program.

http://www.sei.cmu.edu/publications/documents/94.reports/94.sr.001.html

Handbooks

CMU/SEI-94-HB-005
ADA296 788

Description of the Systems Engineering Capability Maturity Model Appraisal Method Version 1.0, A

The purpose of this document is to summarize the major elements of the Systems Engineering Capability Maturity Model (SE-CMM) appraisal method (SAM). (SAM) is a method for using the SE-CMM to benchmark, or otherwise appraise, the process capability of an organization's or enterprise's systems engineering function. The SE-CMM itself is described in SECMM-94-04/CMM-SEI-94-HB-04 [SECMM]. This document describes each step of an SE-CMM appraisal and provides guidance for the preparation and conduct of an appraisal. It also contains background and context information about the appraisal method.

http://www.sei.cmu.edu/publications/documents/94.reports/94.hb.005.html

CMU/SEI-94-HB-004
ADA293345

Systems Engineering Capability Maturity Model, Version 1.0, A

Bate, R.; Garcia, S.; Armitage, J.; Cusick, K.; Jones, R.; Kuhn, D.; Minnich, I.; Pierson, H.; Powell, T.; Reichner, A.

The Systems Engineering Capability Maturity Model (SE-CMM) describes the essential elements of an organization's systems engineering process that must exist to ensure good systems engineering. It does not specify a particular process of sequence. In addition, the SE-CMM provides a reference for comparing actual systems engineering practices against these essential elements. The SE-CMM Model Description provides an overall description of the principles and architecture upon which the SE-CMM is based, an executive overview of the model, suggestions for appropriate use of the model, the practices included in the model, and a description of the attributes of the model. It also includes the requirements used to develop the model.

http://www.sei.cmu.edu/publications/documents/94.reports/94.hb.004.html

CMU/SEI-94-HB-002
ADA286082

Software Capability Evaluation (SCE) Version 2.0 Team Members' Guide

Software Capability Evaluation (SCE) is a method for independently evaluating the software process of an organization to gain insight into its software development capability. The method is defined in the report "Software Capability Evaluation Version 2.0 Method Description."

This document is intended for use by members of teams that will be conducting an SCE. The guide provides detailed step-by-step instructions and heuristic information to assist an SCE team in preparing for and conducting an evaluation.

http://www.sei.cmu.edu/publications/documents/94.reports/94.hb.002.html

CMU/SEI-94-HB-001
ADA285595

Software Process Framework for the SEI Capability Maturity Model, A

Olson, T.; Reizer, N; Over, J.

The Software Process Framework (SPF) is a document that provides information contained in the Software Engineering Institute's Capability Maturity Model (CMM) for Software V. 1. 1 [Paulk93a] in a format suitable for process definition and improvement. The SPF allows users to determine if their organization's software process documentation is consistent with the recommendations made by the CMM. When organizational software process documentation is found to be inconsistent with the CMM, the SPF provides the ability to make informed decisions regarding the applicability of specific CMM recommendations.

http://www.sei.cmu.edu/publications/documents/94.reports/94.hb.001.html

Educational Materials

CMU/SEI-94-EM-011
ADA286083

Rate Monotonic Analysis for Real-Time Systems: Instructor's Guide

Ravenel, R.; Obenza, R.

This educational materials package has been developed for instructors of software engineering and, more specifically, real-time systems. This package will help instructors teach rate monotonic analysis (RMA) to graduate and undergraduate software, computer, and electrical engineering students. The package can also be used to teach RMA to continuing education students. The presentation materials and exercises included have been used by Ruth Ravenel in both graduate and undergraduate courses.

These educational materials are intended to be used in conjunction with the videotape, An Introduction to Rate Monotonic Analysis, from SEI Technology Series. (For instructors who have not already obtained the videotape from the SEI, an order form is included in this package.)

http://www.sei.cmu.edu/publications/documents/ems/94.em.011.html

CMU/SEI-94-EM-10
ADA278536

Lecture Notes on Requirements Elicitation

Raghavan, S.; Zelesnik, G.; Ford, G.

Requirements elicitation is the first of the four steps in software requirements engineering (the others being analysis, specification, and validation). Software engineers use several elicitation techniques. To facilitate teaching these techniques, materials are provided to support an introductory lecture and four lectures on specific techniques: joint application design, brainstorming, interviewing, and the PIECES framework. A role-playing exercise is provided that allows students to experience each of the techniques. Information for instructors includes educational objectives, pedagogical considerations, additional exercises, and a bibliography.

http://www.sei.cmu.edu/publications/documents/ems/94.em.010.html

1993 Reports

Technical Reports

CMU/SEI-93-TR-034
ADA279014

Taxonomy of Coordination Mechanisms Used in Real-Time Software Based on Domain Analysis, A

Fernandez, J.

A taxonomy of the coordination mechanisms for the synchronization and communication of concurrent processes is proposed. The taxonomy deals with the issues of a real-time software architecture that are application domain independent. The taxonomy will help the designer to find the appropriate coordination mechanism for building a real-time domain specific software architecture.

Features Oriented Domain Analysis methodology has been used to describe the taxonomy. While ADA is the programming language that has been used here, some of the attributes and guidelines are still valid for other programing languages.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.034.html

CMU/SEI-93-TR-031
ADA275637

Conceptual Framework for Software Technology Transition, A

Fowler, P.; Levine, L.

We present a conceptual framework that integrates and describes the intersections of three life cycles of software technology transition; research and development, new product development, and adoption and implementation in organizations. We then apply the framework to the technology transition experiences of the Software Engineering Institute.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.031.html

CMU/SEI-93-TR-030

Technology Transition Pull: A Case Study of Rate Monotonic Analysis (Part 2)

Fowler, P.; Levine, L.

This case study reports on efforts to introduce a software technology, rate monotonic analysis, into several software-intensive programs at one site within a multinational firm. We describe lessons learned and success factors in the early use of rate monotonic analysis (RMA).

We also present evidence that supports the requirements for an internal capability—in the form of technical expertise and infrastructure—to adopt and assimilate this new technology. Finally, the study applies the “whole product” concept for understanding technology adoption and use, showing how one form compensated for lack of a whole product in its adoption of RMA.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.030.html

CMU/SEI-93-TR-029
ADA275616

Technology Transition Push: A Case Study of Rate Monotonic Analysis (Part 1)

Fowler, P.; Levine, L.

This case study reports on efforts to transform rate monotonic scheduling theory from an academic theory into a practical analytical technique and to transition that technique into routine practice among developers and maintainers of software embedded in real-time systems.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.020.html

CMU/SEI-93-TR-028
ADA276466

Acquisition Process for the Management of Risks of Cost Overrun and Time Delay Associated with Software Development, An

Haimes, Y.; Chittister, C.

The ability to quantify risk is essential to the process of budgeting and scheduling. During the process of hiring to complete specified tasks, customers must be able to verify contractor estimates and to make sound judgments on the risks of cost overruns and time delays. The following two questions are central to this paper: Do developers with little experience over-estimate or under-estimate the complexity of the task because of their past experience, the assumptions they make, the models they select, and how they define the model parameters? What are the sources of risk associated with project cost estimation? How can such risk be quantified? To address these questions, this paper proposed a systematic acquisition process that is aimed at assessing and managing the risks of cost overruns and time delays associated with software development.

The proposed acquisition process, which is composed of four phases (listed below), is grounded on the following three basic premises: a) Any single-value estimate of cost or completion time is inadequate to capture and represent the variability and uncertainty associated with cost and schedule. Probabilistic quantification is advocated, using, in this paper, the fractile method and triangular distribution. b) The common expected value when used as a measure of risk, is inadequate; further, if used as the sole measure of risk, it may lead to inaccurate results. The conditional expected value of risk of extreme events is adopted to supplement and complement the common unconditional expected value. c) Probing the sources of risks and uncertainties associated with cost overruns and time delays in software development is essential for the ultimate management of technical and nontechnical risks. The Taxonomy-Based Questionnaire developed by the Software Engineering Institute is adopted.

These basic premises have led to the development of the following four phases in the proposed acquisition process: Phase I, constructing the probability density functions; Phase II, probing the sources of risks and uncertainties; Phase II, analyzing and regarding the likelihood of technical and non-technical risks; and Phase IV, drawing conclusions on the basis of the accumulated evidence and ultimately selecting the contractors most likely to complete the project without major cost overruns or time delays. The three example problems are presented to demonstrate the construction of the probability density functions in Phase I and to explain in a more general way the effort involved in Phases II through IV.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.028.html

CMU/SEI-93-TR-027
ADA278595

Survey of Commonly Applied Methods for Software Process Improvement, A

Austin, R.; Paulish, D.

This report describes a number of commonly applied methods for improving the software development process. Each software process improvement method is described by surveying existing technical literature citations. Each method description contains background information concerning how the method works. Documented experience with the method is described. Suggestions are given for implementing the method, and a list of key references is given for further information. The methods are described in the context of the SEI Capability Maturity Model, and suggestions are given to assist organizations in selecting potential improvement methods based upon their current process maturity.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.027.html

CMU/SEI-93-TR-026
ADA277289

Case Studies of Software Process Improvement Methods

Paulish, D.

This report describes the case studies approach applied at a number of Siemens software development organizations to observe the impact of software process improvement methods. In addition, the report provides guidance to software development organizations that want to improve their processes. A set of organization performance measures are defined to help an organization observe its software process improvement over time. An approach is given for selecting software process improvement methods. The report concludes with a description of common implementation problems, and recommendations for organizations to improve their software processes.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.026.html

CMU/SEI-93-TR-025
ADA263432

Key Practices of the Capability Maturity Model Version 1.1

Paulk, M.; Weber, C.; Garcia, S.; Chrissis, M.; Bush, M.

This document provides the key practices that correspond to each maturity level of the Capability Maturity Model and information on how to interpret the key practices. It is an elaboration of what is meant by maturity at each level of the CMM and a guide that can be used for software process improvement, software process assessments, and software capability evaluations.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.025.html

CMU/SEI-93-TR-024
ADA263403

Capability Maturity Model for Software (Version 1.1)

Paulk, M.; Curtis, B.; Chrissis, M.; Weber, C.

In November 1986, the Software Engineering Institute (SEI) with assistance from the Mitre began developing a process maturity framework that would assist organizations in improving their software process. This effort was initiated in response to a request to provide the federal government with a method for assessing the capability of their software contractors. In September 1987, the SEI released a brief description of the process maturity framework and a maturity questionnaire (CMU/SEI-87-TR-023). The SEI intended the maturity questionnaire to provide a simple tool for identifying areas where an organization's software process needed improvement. Unfortunately, the questionnaire was too often regarded as "the model" rather than as a vehicle for exploring process maturity issues.

After four years of experience with the software process maturity framework and the preliminary version of the maturity questionnaire, the SEI has evolved the software process maturity framework into a fully defined model.

This model will be used in a systematic, principled way to derive a maturity questionnaire. By fully elaborating the maturity framework, a model has emerged that provides organizations with more effective guidance for establishing process improvement programs than was offered by the maturity questionnaire. Using knowledge acquired from software process assessments and extensive feedback from both industry and government, an improved version of the process maturity framework has been produced called the Capability Maturity Model for Software (CMM). This paper is an introduction to the revised model.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.024.html

CMU/SEI-93-TR-023
ADA275169

Reference Model for Project Support Environments (Version 2.0)

The U.S. Navy has embarked on the next Generation Computer Resources (NGCR) program to fulfill its need for standard computing resources. The program revolves around the selection of interface standards in six areas. The interface standards will be based on existing industry standards with multi-vendor support. The objective is to restructure the Navy's approach to take better advantage of commercial advances and to reduce cost and duplication of computer resources. This document is part of the NGCR program.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.023.html

CMU/SEI-93-TR-019
ADA277169

ADA Binding to the SAFENET Lightweight Application Services, An

Meyers, B.; Chastek, G.

This document describes an ADA binding to the Survivable ADAptable Fiber Optic Embedded Network (SAFENET) lightweight application services. The major goal in the design of the binding was schedulability. The document contains the ADA package specifications for the binding as well as a rationale for the design.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.019.html

CMU/SEI-93-TR-018
ADA269922

Software Capability Evaluation (SCE) Version 1.0 Implementation Guide

Software Capability Evaluation (SCE) offers a means to evaluate an organization's software process capability—that is, how well an organization manages the process it used to create software. SCE provides a way to compare an offeror's software capability against a predefined standard. This document is an implementation guide: it is intended as a set of practical information which program managers can use to guide them through the process of using SCE in an acquisition.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.018.html

CMU/SEI-93-TR-017
ADA267895

Software Capability Evaluation (SCE) Version 1.5 Method Description

Averill, E.; Byrnes, P.; Dedolph, M.F.; Maphis, J.; Mead, W.; Puranik, R.

This report describes Version 1.5 of the Software Capability Evaluation (SCE) method, as taught at the Software Engineering Institute (SEI) from January 1992 to June 1993. This version of the SCE method is based on the process maturity framework defined in *Characterizing the Software Process: A Maturity Framework* (CMU/SEI-87-TR-011), which predates Version 1.0 of the Capability Maturity Model (CMM) described in *Capability Maturity Model for Software* (CMU/SEI-91-TR-024). The document includes a background overview of the SCE method and its evolution, a detailed description of the activities performed during an SCE, and a discussion of the characteristics of the method and their implications for the use of the method. This "as built" description provides a baseline for future evolution of the SCE method.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.017.html

CMU/SEI-93-TR-016
ADA267896

Establishing a Software Measurement Process

McAndrews, D.

This report presents guidelines for establishing a measurement process as part of an organization's overall software process. Methods are suggested that can be used to design a repeatable measurement process that is focused on goal setting, data analysis, and decision making rather than on just data collection and numbers. Examples are included to illustrate these methods in the context of some common software process management issues involving basic measures of size, effort, and schedule. This report also suggests some steps for starting a measurement program. These steps focus on identifying an organizational strategy for improvement and designing a process for measurement to support this strategy.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.016.html

CMU/SEI-93-TR-015
ADA272441

Investigation into the State of the Practice of CASE Tool Integration, An

Rader, J.; Brown, A.; Morris, E.

In the second half of 1992 a team of the SEI CASE Environments Project conducted a study into the state of the practice with respect to the operational use of integrated Computer-Aided Software Engineering (CASE) tools. After many false starts, we interviewed a number of examples of large organizations with integrated CASE tools in operational use on software development projects. Compared to the state of the art described in much of the literature, what was found might be considered modest. Compared to industry norms, it was quite impressive, representing significant commitment, ingenuity, and significant attention to end user needs.

This report details our observations and analyzes the current state of the practice of CASE tool integration as revealed by our study. It also speculates on reasons for the modest state of the practice.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.015.html

CMU/SEI-93-TR-014
ADA271348

Structural Modeling: An Application Framework and Development Process for Flight Simulators

Abowd, G.; Bass, L.; Howard, L.; Northrop, L.

In this paper, we present the structural modeling approach, an application framework and development process for the construction of flight simulators. Structural modeling was developed to address functional, nonfunctional, and process requirements for flight simulators. It has been successfully used in the development of large scale (one million lines of ADA code) flight simulators for the United States Air Force. A structural model promotes a simple and coherent software architecture with a small number of specialized structural elements obeying a few systemwide coordination strategies. It is this simplicity coherence of the software architecture that enables analysis to demonstrate the quality of the system.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.014.html

CMU/SEI-93-TR-013
ADA272570

Software Product Liability

Armour, J.; Humphrey, W.

Voyne Ray Cox settled into the radiation machine for the eighth routine treatment of his largely cured cancer. The operator went to the control room and pushed some buttons. Soon, the machine went into action and the treatment began. A soft whir and then an intense searing pain made him yell for help and jump from the machine. The doctors assured him there was nothing to worry about. What they didn't know was that the operator had inadvertently pushed an unusual sequence of controls that activated a defective part of the software controlling the machine. He didn't die for six months but he had received a lethal dose of radiation. This software defect actually killed two patients and severely injured several others. The final decisions in the resulting lawsuits have not been made public.

Software defects are rarely lethal and the number of injuries and deaths is now very small. Software, however, is now the principle controlling element in many industrial and consumer products. It is so pervasive that it is found in just about every product that is labeled "electronic." Most companies are in the software business whether they know it or not. The question is whether their products could potentially cause damage and what their exposures would be if they did.

While most executives are now concerned about product liability, software introduces a new dimension. Software, particularly poor quality software, can cause products to do strange and even terrifying things. Software bugs are erroneous instructions and, when computers encounter them, they do precisely what the defects instruct. An error could cause a 0 to be read as a 1, an up control to be shut down, or, as with the radiation machine, a shield to be removed instead of inserted. A software error could mean life or death.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.013.html

CMU/SEI-93-TR-012
ADA268058

AMORE: The Advanced Multimedia Organizer for Requirements Elicitation

Christel, M.; Wood, D.; Stevens, S.

The advent of larger and more complex software systems has resulted in the need to reconsider the ways in which information pertaining to those systems is stored, visualized, organized, and retrieved. The Software Engineering Information Modeling Project of the Software Engineering Institute is integrating existing and new technologies with the intent of demonstrating feasible technical directions for modeling and managing large amounts of data in multiple media formats. This paper introduces the Advanced Multimedia Organizer for Requirements Elicitation (AMORE), a system that embodies a synthesis of technologies adapted specifically for application to requirements elicitation processes and models.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.012.html

CMU/SEI-93-TR-011
ADA280940

Integrating 001 Tool Support into the Feature-Oriented Domain Analysis Methodology

Krut Jr., R.

This report addresses the need for additional tool support for the Feature-Oriented Domain Analysis (FODA) methodology, developed at the Software Engineering Institute (SEI). Previous FODA studies relied on multiple tools to represent the components of a domain model. This report discusses the ability to represent an analyzed domain within the confines of a single support tool. This discussion was based on the transformation of a recently completed domain analysis from a multi-tool, multi-view representation into a single tool which represents the multiple views of a FODA domain model. This report also describes the potential for prototyping of systems using the FODA domain analysis products and the supporting tool.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.011.html

CMU/SEI-93-TR-010
ADA273769

Distribution Systems, The

Meyers, B.; Chastek, G.

This report provides an overview of two standards that are used for data specification and representation in distributed systems. The standards considered are the Abstract Syntax Notation One (ASN.1) and the external data representation (XDR). Standards for data representation are appropriate for the development of real-time distributed systems, particularly loosely coupled, heterogeneous systems. The report presents an example of the use of each standard. Several performance metrics are also introduced that are suitable for comparing the space and time costs of using the different standards. Several issues are discussed that are appropriate to a system designer. An ADA implementation of ASN.1 encode and decode routines for floating point types is included in an appendix.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.010.html

CMU/SEI-93-TR-009
ADA266994

Concepts on Measuring the Benefits of Software Process Improvement

Rozum, J.

The software community initially became aware of process improvement from the works of Deming, Juran, and Crosby. The current awareness and activity regarding software process improvement was sparked by the Software Engineering Institute (SEI) with the release of its original software process maturity model. Following the advice of the SEI, many software organizations initiated software process improvement efforts to improve the quality of their products by improving the processes that produce those products. The question that many managers are continually asking today, though, is: How much has my organization benefited from the changes we have made? Unfortunately, many organizations did not include a method of measuring those benefits in their improvement activities. This report describes some concepts that organizations can tailor and build upon to develop a method for determining the benefits they have received from their software process improvement activities.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.009.html

CMU/SEI-93-TR-008
ADA269923

Study in Software Maintenance, A

Dart, S.; Christie, A.; Brown, A.

In an effort to find out more about the tools, procedures, and techniques project personnel use in their work, the Computer-Aided Software Engineering (CASE) Environments Project interviewed personnel in eight software maintenance projects within an agency of the U.S. government. These interviews highlighted problems that we believe are typical of many software maintenance organizations (i.e., the need for more effective software maintenance tools, lack of communication between individuals working on similar projects, low status of maintenance personnel, and lack of a design-for-maintenance philosophy during the software development phase). This report highlights the findings of these interviews, provides our analysis of the findings, and makes recommendations directed at the agency for improvement in the areas of tools, people, and process. We believe that what we observed is very typical of the state of the practice in these areas and as such that this report and its recommendations are applicable to other large or small software maintenance projects.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.008.html

CMU/SEI-93-TR-007
ADA280916

Establish a Software Measurement Program The SEI and NAWC: Working Together to Establish a Software Measurement Program

Rozum, J.

In 1990, the Software Engineering Institute (SEI) and the then Naval Air Development Center (NADC) in Warminster, Pennsylvania (now Naval Air Warfare Center, Aircraft Division, Warminster) signed an agreement to jointly develop a software measurement program for NADC - Warminster. To help with that development, a software measurement process action team (SMPAT) was formed with members from the SEI and NADC. The SMPAT's responsibility was to plan, develop, and assist in the implementation of the measurement program. The purpose of this technical report is to document and make available to the software community the process and methods used, experience gained, and some lessons learned in establishing the software measurement program at NADC.

This report is meant to help organizations that desire to start a software measurement program or have been struggling with such a program by providing an example of one organization that has also struggled to establish a software measurement program. To help an organization, real-life examples of how software measures were defined, collected, and used to improve the management process are included.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.007.html

CMU/SEI-93-TR-006
ADA266992

Taxonomy-Based Risk Identification

Carr, M.; Konda, S.; Monarch, I.; Ulrich, F.; Walker, C.

This report describes a method for facilitating the systematic and repeatable identification of risks associated with the development of a software-dependent project. This method, derived from published literature and previous experience in developing software, was tested in active government-funded defense and civilian software development projects for both its usefulness and for improving the method itself. Results of the field tests encouraged the claim that the described method is useful, usable, and efficient. The report concludes with some macro-level lessons learned from the field tests and a brief overview of future work in establishing risk management on a firm footing in software development projects.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.006.html

CMU/SEI-93-TR-005
ADA266993

Safety-Critical Software: Status Report and Annotated Bibliography

Place, P.; Kang, K.

Many systems are deemed safety-critical and these systems are increasingly dependent on software. Much has been written in the literature with respect to system and software safety. This report summarizes some of that literature and outlines the development of safety-critical software. Techniques for hazard identification and analysis are discussed. Further, techniques for the development of safety-critical software are mentioned. A partly annotated bibliography of literature concludes the report.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.005.html

CMU/SEI-93-TR-004
ADA26103

Process-Centered Development Environments: An Exploration of Issues

Christie, A.

Software development environments are beginning to move from research communities to commercial applications. As this occurs, the need to address process issues related to such environments is becoming increasingly apparent. Thus there is a growing awareness of the need for process-centered development environments (PCDEs). This report addresses process definition and enactment issues which pertain to the specification and design of a PCDE. The first part of the report explores some of the required characteristics of an enactable graphical language and the relationship between process definition and enactment. This process language naturally led to the ability to perform process verification, i.e., a verification that the actual process path taken throughout a project conforms to the defined process. The issue of process verification is thus also explored. The success of PCDEs rests heavily on end-user acceptance. Because of this, the report concludes with a review of user-oriented process and social issues relevant to the successful adoption of PCDEs.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.004.html

CMU/SEI-93-TR-003
ADA266995

Software Architectures for Shared Information Systems

Shaw, M.

Software system design takes place at many levels. Different kinds of design elements, notations, and analyses distinguish these levels. At the *software architecture* level, designers combine subsystems into complete systems. This paper studies some of the common patterns, or idioms, that guide these configurations. Results from software architecture offer some insight into the problems of systems integration—the task of connecting individual, isolated, pre-existing software systems to provide coherent, distributed solutions to large problems. As computing has become more sophisticated, so too have the software structures used in the integration task. This paper reviews historical examples of shared information systems in three different applications whose requirements share some common features about collecting, manipulating, and preserving large bodies of complex information. These applications have similar architectural histories in which a succession of designs responds to new technologies and new requirements for flexible, highly dynamic responses. A common pattern, the shared *information system evolution pattern*, appears in all three areas.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.003.html

CMU/SEI-93-TR-002
ADA226519

Distributed Real-Time System Design: Theoretical Concepts and Applications

Sha, L.; Sathaye, S.

Distributed real-time system design raises new theoretical issues and application challenges, beyond those of centralized systems. Rate monotonic scheduling (RMS) theory has been successfully applied in the scheduling of centralized systems. RMS and its generalizations have been adopted by national high technology projects such as the Space Station and has recently been supported by major open standards such as the IEEE Futurebus+ and POSIX.4. In this paper, we describe the use of generalized rate monotonic scheduling theory for the design and analysis of a distributed real-time system. We review the recent extensions of the theory to distributed system scheduling, examine the architectural requirements for use of the theory, and finally provide an application example.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.002.html

CMU/SEI-93-TR-001
ADA265202

Overview of PCTE: A Basis for a Portable Common Tool Environment, An

Long, F.; Morris, E.

Environment framework technologies are becoming increasingly popular as aids to the construction of software engineering environments populated with integrated CASE tools. PCTE, a framework technology, is generating significant interest among environment users and builders as a mechanism for data management. This report details the history and current status of PCTE and PCTE-based environments. The major capabilities of PCTE are identified, and the techniques for integration of tools into PCTE are discussed.

http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.001.html

Special Reports

CMU/SEI-93-SR-021
ADA275642

Process Guide for the Domain-Specific Software Architectures (DSSA) Process Life Cycle

Armitage, J.

This document describes the prototype domain-specific software architecture (DSSA) process life cycle developed by GTE as part of the ARPA, formerly DARPA DSAA program. It is a high-level process description and represents a snapshot of the process as it was in the fall of 1992.

http://www.sei.cmu.edu/publications/documents/93.reports/93.sr.021.html

CMU/SEI-93-SR-020

Results of a Workshop on Research in Incident Handling

Longstaff, T.

This document contains the results of the first CERT^{SM} Invitational Workshop on Research in Incident Handling, held at the Software Engineering Institute in November 1992. The workshop was convened to address a wide spectrum of computer, network, and information security topics from the perspective of incident handling, both in the present and in the future. The intent was to bring together researchers, incident handling specialists, users, system administrators, and managers to encourage an exchange of information and experience. Specifically, it was intended to identify lucrative areas for research and development in improving the practice of incident handling and in applying the experience- and information-base that the CERT Coordination Center has amassed during its existence.

http://www.sei.cmu.edu/publications/documents/93.reports/93.sr.020.html

CMU/SEI-93-SR-007
ADA268059

Software Process Framework for the SEI Capability Maturity Model: Repeatable Level, A

Olson, T.; Parker Gates, L.; Mullaney, J.; Over, J.; Reizer, N.; Kellner, M.; Phillips, R.; DiGennaro, S.

This document describes a Software Process Framework (SPF) based on the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) for the Repeatable Level (Level 2). The purpose of the SPF is to provide guidance for designing, analyzing, and reviewing software processes (consisting of standards, processes, procedures, training, and tools) so that they are consistent with the CMM. As a thesaurus is a companion book to a dictionary (both are useful for different purposes) the SPF is a companion guide to the CMM for defining software processes. For each key process area in the CMM, the SPF defines roles, entry and exit criteria, inputs and outputs, activities, reviews and audits, measurements, etc. This information can also be used to help build process models and process guides that are consistent with the CMM. The primary audiences of the SPF are software engineering process groups (SEPGs), process engineers, process action teams, and software quality assurance groups. Secondary audiences include anyone interested in software process improvement, or anyone using the CMM.

http://www.sei.cmu.edu/publications/documents/93.reports/93.sr.007.html

CMU/SEI-93-SR-005
ADA267117

Reengineering: An Engineering Problem

Feiler, P.H.

This paper discusses a plan that addresses how the Software Engineering Institute (SEI) may assist the Department of Defense (DoD) in reengineering its large software-intensive systems. This plan is based on a view of reengineering as an engineering problem to improve the cost-effective evolution of large software-intensive systems. This view of reengineering, which takes the whole software engineering process into account, fosters a growth path by leveraging promising emerging software engineering technologies. Reengineering also builds on the industry's improvement in its ability to manage the software engineering process, an accomplishment of SEI work in the Capability Maturity Model (CMM) and its key process areas.

http://www.sei.cmu.edu/publications/documents/93.reports/93.sr.005.html

CMU/SEI-93-SR-004
ADA267103

Dependable Software Technology Exchange

Weinstock, C.; Schneider, F.B.

On March 18 and 19, 1993, the Dependable Real-Time Software Project hosted a Dependable Software Technology Exchange. The exchange, sponsored by the Air Force Space and Missile Systems Center and the Office of Naval Research, brought together researchers and system developers, providing an opportunity for the researchers to learn the needs of the developers and for the developers to learn about techniques being investigated by the researchers. This report summarizes what transpired at the meeting.

http://www.sei.cmu.edu/publications/documents/93.reports/93.sr.004.html

Curriculum Modules and Support Materials

SEI-CM-27-1.0
ADA265201

Formal Specification and Verification of Concurrent Programs

Berry, D.

February 1993

This module introduces formal specification of concurrent software and verification of the consistency between concurrent programs and their specifications. First, what one might want to be able to prove about a concurrent program is discussed. Then, a number of formal descriptions of the concept are presented. These vary in their coverage of the phenomena, and some can be used as the bases of formal specification of programs. Next, techniques for carrying out the proof of consistency between the specification and the program are described. Finally, it is noted that some of the these techniques have automated tools such as verifiers associated with them.

http://www.sei.cmu.edu/publications/documents/cms/cm.027.html

Educational Materials

CMU/SEI-93-EM-009
ADA266959

Lecture Notes on Engineering Measurement for Software Engineers

Ford, G.

Measurement is a fundamental skill for engineers. To facilitate teaching software engineering measurement, materials are provided to support three lectures: introduction to engineering measurement, measurement theory, and software engineering measures. These materials include lecture notes suitable for class handouts and additional information for instructors—educational objectives, pedagogical considerations, suggestions for class projects, an annotated bibliography, and transparency masters for use in the delivery of the lectures.

http://www.sei.cmu.edu/publications/documents/ems/93.em.009.html

CMU/SEI-93-EM-008
ADA265200

Lecture Notes on Software Process Improvement

Werth, L.

Software process improvement is not usually covered in standard software engineering textbooks. However, because it is a topic of great interest to the software industry, both faculty and students should be familiar with it. The goal of this package is to provide the basis for an introductory 30 to 60 minute lecture on the software process and its improvement.

http://www.sei.cmu.edu/publications/documents/ems/93.em.008.html

CMU/SEI-93-EM-007
ADA264273

Materials for Teaching Software Inspections

Tomayko, J.; Murphy, J.

This educational materials package was developed for instructors of software verification techniques in graduate and undergraduate software engineering courses, and for those who teach industrial continuing education courses on the meaning and methods of software inspections.

Software inspections are a low-tech, highly effective verification technique. Research has consistently shown that the defect detection rate of inspections is higher than that of many traditional testing techniques. This package includes materials for demonstrating how to perform an inspection and also for "selling" students on the effectiveness of inspections. It complements EM-5 "Scenes of Software Inspections," providing additional background material and exercises for using that set of educational materials.

http://www.sei.cmu.edu/publications/documents/ems/93.em.007.html

1992 Reports

Technical Reports

CMU/SEI-92-TR-036
ADA275345

Durra: A Task Description Language User's Manual (Version 2)

Doubleday, D.; Barbacci, M.

This document describes the use of Durra, a task-level application description language, and its associated toolset. The Durra environment supports the development of highly reconfigurable distributed ADA applications. The intended audience for this document is system managers responsible for Durra installation and Durra application developers.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.036.html

CMU/SEI-92-TR-035
ADA259853

Control Integration through Message Passing

Brown, A.

Understanding tool integration in a Software Development Environment (SDE) is one of the key issues being addressed in current work on providing automated support for large-scale software production. Work has been taking place at both the conceptual level ("What is integration?") and the mechanistic level ("How do we provide integration?"). Many people see the answers to these questions as providing the cornerstone of real progress in the area.

Until recently, existing integration mechanisms have been very rigid in the support for integration that they provide. Users have been offered a fixed level of integration with little flexibility. However, one approach that has been recently implemented employs a control integration paradigm that appears to be flexible, supportive, and adaptable to a wide range of end-user needs. Implementations of this paradigm are based on the notion of "message passing" as the underlying communication mechanism between SDE services.

In this paper we examine the message passing approach to integration in an SDE, look at the general principles of the approach, describe some existing implementations, and discuss the use of such a mechanism as the basis for a more flexible environment that is open to experimentation with different approaches to integration.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.035.html

CMU/SEI-92-TR-034
ADA260241

Academic Legitimacy of the Software Engineering Discipline

Berry, D.

This report examines the academic substance of software engineering. It identifies the basic research questions and the methods used to solve them. What is learned during this research constitutes the body of knowledge of software engineering. The report then discusses at length what about software makes its production so difficult and makes software engineering so challenging an intellectual discipline.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.034.html

CMU/SEI-92-TR-033
ADA258457

Conceptual Framework for System Fault Tolerance, A

Heimerdinger, W.; Weinstock, C.

A major problem in transitioning fault tolerance practices to the practitioner community is a lack of a common view of what fault tolerance is, and how it can help in the design of reliable computer systems. This document takes a step towards making fault tolerance more understandable by proposing a conceptual framework. The

framework provides a consistent vocabulary for fault tolerance concepts, discusses how systems fail, describes commonly used mechanisms for making systems fault tolerant, and provides some rules for developing fault tolerant systems.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.033.html

CMU/SEI-92-TR-032
ADA264375

Performance and ADA Style for the AN/BSY-2 Submarine Combat System

Altman, N.; Donohoe, P.

The performance of programs prepared with the Verdix ADA Development System (VADS) was measured and analyzed for programmers preparing a large ADA system. Using standard ADA benchmark suites (ACEC, AES and PIWG) and a representative Motorola 68030 target system as a source of data, questions were posed and answered about programming alternatives, based on the measured performance of the compiler. The questions included in the report were extracted from a much larger set selected from an analysis of the BSY-2 Style Guide and augmented with additional questions suggested by SEI experience. The derivation of the questions and the template for the performance analysis sections are presented as appendices.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.032.html

CMU/SEI-92-TR-031
ADA258459

Analysis of a Software Maintenance System: A CASE Study

Slomer, H.; Christie, A.

To design, implement, and operate a successful software development process, exposure to similar existing systems is invaluable. The objective of this paper is thus to document and analyze an existing, moderate size, software maintenance project. The project, which supports the maintenance of a software environment has, through incremental improvement, become very effective. However, this effectiveness has only been achieved through struggle, compromise, and creativity. The paper documents the evolution of the project, providing insights into how change was managed, and defines and formally models the project as it existed until recently. The project's process is still evolving, and recent changes, while not formally modeled, are also described. The results of this modeling are applied 1) to compare the project's practices from a perspective of the SEI Capability Maturity Model (CMM), and 2) to address briefly the issue of process reuse. Comparison to the CMM resulted in an identification of strengths and weaknesses of the project's software process. In the examination of reuse issues, three hypothetical examples of process reuse are examined.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.031.html

CMU/SEI-92-TR-030
ADA258743

Software Development Risk: Opportunity, Not Problem

Van Scoy, R.

What is risk? What is risk management? What does risk management have to do with software? Noted software expert Tom Gilb says:

> If you don't actively attack the risks, they will actively attack you.
>
> — [Gilb88, p. d72]

But what does it mean to actively attack risks? We answer these questions by examining the problems that exist in software development today and presenting the SEI Risk Program approach to turning risk into opportunity.

We begin by reviewing the fundamental concepts of risk and elaborating on how these basic concepts apply to the development of large, software-intensive systems. We then develop our strategy for seeing a systematic approach to risk management in software development be routinely practiced.

There are two key activities we are using to implement our strategy. The first is our risk management paradigm. The paradigm defines a set of continuous activities that must be undertaken to resolve technical risk in a systematic and structured way. The second is our risk assessment process for collaborating with clients to identify their technical risks.

We end with our ultimate goal: establishing an effective risk management ethic as standard practice in the software engineering industry.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.030.html

CMU/SEI-92-TR-029
ADA258937

ADA Adoption Handbook: A Program Manager's Guide Version 2.0

Hefley, W.; Foreman, J.; Engle, Jr., C.; Goodenough, J.

The *ADA Adoption Handbook* provides program managers with information about how best to tap ADA's strengths and manage the transition to fully using this software technology. Although the issues are complex, they are not all unique to ADA. Indeed, many of the issues addressed in this handbook must be addressed when developing any software-intensive system in any programming language. The handbook addresses the advantages and risks in adopting ADA. Significant emphasis has been placed on providing information and suggesting methods that will help program and project managers succeed in using ADA across a broad range of application domains.

The handbook focuses on the following topics: ADA's goals and benefits; program management issues; implications for education and training; software tools with emphasis on compiler validation and quality issues; the state of ADA technology as it related to system design and implementation; and the pending update of the ADA language standard (ADA 9X).

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.029.html

CMU/SEI-92-TR-024
ADA266996

Analysis of SEI Software Process Assessment Results 1987-1991, An

Kitson, D.; Masters, S.

This report focuses on the results of SEI software process assessments conducted over a four year period beginning in 1987. It characterizes the software processes used by software managers and practitioners at the assessed sites and classifies issues identified during the assessments. The basis for the characterization and classification is a software process maturity model developed by the SEI. This report contributes to the existing body of knowledge on the state of practice of software engineering in the U.S. by characterizing the sites from a software process maturity perspective and profiling site software process weaknesses. The data analyzed is drawn from SEI software process assessments of 59 government and industry software sites. This work is an analysis of existing assessment data rather than a designed study. The participating sites were not randomly selected; accordingly, they do not necessarily constitute a statistically valid sampling of the U.S. software industry.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.024.html

CMU/SEI-92-TR-023
ADA258254

Concept Study for a National Software Engineering Database, A

Van Verth, P.

Substantial segments of the software engineering community perceive a need for high-quality software engineering data at a national level. A national software engineering database has been proposed as one way to satisfy this need. But is such a database feasible and can it really satisfy these needs?

This report provides information obtained from an informal survey of members of the software engineering community about a national database. The survey served as a means of getting a sense of the expectations that are to be met by building a national database, and provided the opportunity to learn from the experiences of those surveyed about data collection and use. The report summarizes this material in a manner that is informative and expository rather than prescriptive.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.023.html

CMU/SEI-92-TR-022
ADA258556

Software Quality Measurement: A Framework for Counting Problems and Defects

Florac, W.

This report presents mechanisms for describing and specifying two software measures—software problems and defects—used to understand and predict software product quality and software process efficacy. We propose a framework that integrates and gives structure to the discovery, reporting, and measurement of software problems and defects found by the primary problem and defect finding activities. Based on the framework, we identify and organize measurable attributes common to these activities. We show how to use the attributes with checklists and supporting forms to communicate the definitions and specifications for problem and defect measurements. We illustrate how the checklist and supporting forms can be used to reduce the misunderstanding of measurement results and can be applied to address the information needs of different users.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.022.html

CMU/SEI-92-TR-021
ADA258279

Software Effort and Schedule Measurement: A Framework for Counting Staff-Hours and Reporting Schedule Information

Goethert, W.; Bailey, E.; Busby, M.

This report and the methods in it are outgrowths of work initiated by the Effort and Schedule Subgroup of the Software Metrics Definition Working Group. It contains guidelines and advice from software professionals. It is not a standard, and it should not be viewed as such. Nevertheless, the frameworks and recommendations it presents give a solid basis for constructing and communicating clear definitions for some important measures that can help all of us plan, manage, and improve our software projects and processes.

We hope that the materials we have assembled will give you a solid foundation for making your effort and schedule measures repeatable, internally consistent, and clearly understood by others. We also hope that some of you will take the ideas illustrated in this report and apply them to other measures, for no single set of measures can ever encompass all that we need to know about software products and processes.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.021.html

CMU/SEI-92-TR-020
ADA258304

Software Size Measurement: A Framework for Counting Source Statements

Park, R.

This report presents guidelines for defining, recording, and reporting two frequently used measures of software size—physical source lines and logical source statements. We propose a general framework for constructing size definitions and use it to derive operational methods for reducing misunderstandings in measurement results. We show how the methods can be applied to address the information needs of different users while maintaining a common definition of software size.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.020.html

CMU/SEI-92-TR-019
ADA258305

Software Measurement for DOD Systems: Recommendations for Initial Core Measures

Carleton, A.; Park, R.; Goethert, B.; Florac, A.; Bailey, E.; Pfleeger, S.

This report presents our recommendations for a basic set of software measures that Department of Defense (DoD) organizations can use to help plan and manage the acquisition, development, and support of software systems. These recommendations are based on work that was initiated by the Software Metrics Definition Working Group and subsequently extended by the SEI to support the DoD Software Action Plan. The central theme is the use of checklists to create and record structured measurement descriptions and reporting specifications. These checklists provide a mechanism for obtaining consistent measures from project to project and for communicating unambiguous measurement results.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.019.html

CMU/SEI-92-TR-017
ADA258221

Experience with a Course on Architectures for Software Systems Part I: Course Description

Garlan, D.; Shaw, M.; Okasaki, C.; Scott, C.; Swonger, R.

As software systems grow in size and complexity their design problem extends beyond algorithms and data structures to issues of system design. This area receives little or no treatment in existing computer science curricula. Although courses about specific systems are usually available, there is no systematic treatment of the organizations used to assemble components into systems. These issues—the software architecture level of software design—are the subject of a new course that we taught for the first time in spring 1992. In this pair of reports, Part I presents the motivation for the course, the content and structure of the current version, and our plans for improving the next version. Part II consists of teaching materials from the first offering, including assignments and overheads for lectures.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.017.html

CMU/SEI-92-TR-016
ADA258758

Rationale for SQL ADA Module Language Description (SAMeDL)

Chastek, G.; Graham, M.; Zelesnik, G.

The SQL ADA Module Description Language, SAMeDL, is a language for the specification of Abstract Interfaces as delineated by the SQL ADA Module Extensions (SAME) methodology. The language is formally defined in the SAMeDL Reference Manual [Chastek]. This document is a companion to the Reference Manual. Whereas the Reference Manual is meant to be precise, the Rationale is meant to be clear.

An explanation of the problem solved by the SAMeDL is given. The creation of a new language is justified and the underlying principles of that language are described. Crucial issues in the language are then explained. These include:

- The form and meaning of identifiers in the SAMeDL.
- The role of and procedures for data definition in the SAMeDL. This includes support for enumerations and constants in the SAMeDL.
- The typing rules of the SAMeDL.
- The proposed use of some SAMeDL features is also illustrated. These include Standard Post Processing and User Defined Base Domains.

This document is a revision of an earlier technical report, SEI/CMU-91-TR-4. The revision consists of the addition of a reference model of database programming language interfaces to Section 1.3. The reference model gives a context for the survey of ADA SQL interface solutions given in Section 1.3.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.016.html

CMU/SEI-92-TR-015
ADA258852

Guide to CASE Adoption

Stepien Oakes, K.; Smith, D.; Morris, E.

In an attempt to address the productivity and quality problems afflicting the software industry, many organizations are turning toward computer-aided software engineering (CASE) technology as a potential solution. Unfortunately, the inflated claims of vendors and unreasonable expectations of new users have led to many failed CASE adoption efforts. This guide answers questions organizations may have concerning CASE technology, and provides a strategy for the adoption of CASE tools into an organization.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.015.html

CMU/SEI-92-TR-013
ADA258466

Classification and Bibliography of Software Prototyping, A

Wood, D.; Kang, K.

Prototyping, the creation and enaction of models based on operational scenarios, has been advocated as a useful software engineering paradigm because it lends itself to intense interaction between customers, users, and developers, resulting in early validation of specifications and designs. An extensive and widespread interest in software prototyping in recent years has resulted in a daunting amount of literature and dozens of proposed methods and tools. As with any immature and growing technology, the expanding literature and approaches have resulted in correspondingly expansive and confusing terminology. This report presents an overview of technology and literature relating to the creation and use of software system prototypes. In addition to an annotated bibliography of recent prototyping literature, a technology framework, taxonomy, and series of classifications are provided. The intent of this report is to provide a basic road map through the available literature and technology.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.013.html

CMU/SEI-92-TR-012
ADA258932

Issues in Requirements Elicitation

Christel, M.; Kang, K.

There are many problems associated with requirements engineering, including problems in defining the system scope, problems in fostering understanding among the different communities affected by the development of a given system, and problems in dealing with the volatile nature of requirements. These problems may lead to poor requirements and the cancellation of system development, or else the development of a system that is later judged unsatisfactory or unacceptable, has high maintenance costs, or undergoes frequent changes. By improving requirements elicitation, the requirements engineering process can be improved, resulting in enhanced system requirements and potentially a much better system.

Requirements engineering can be decomposed into the activities of requirements elicitation, specification, and validation. Most of the requirements techniques and tools today focus on specification, i.e., the representation of the requirements. This report concentrates instead on elicitation concerns, those problems with requirements engineering that are not adequately addressed by specification techniques. An elicitation methodology is proposed to handle these concerns.

This new elicitation methodology strives to incorporate the advantages of existing elicitation techniques while comprehensively addressing the activities performed during requirements elicitation. These activities include fact-finding, requirements gathering, evaluation and rationalization, prioritization, and integration. Taken by themselves, existing elicitation techniques are lacking in one or more of these areas.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.012.html

CMU/SEI-92-TR-011
ADA254177

Software Measurement Concepts for Acquisition Program Managers

Rozum, J.

For program managers to effectively manage and control software development, they need to incorporate a measurement process into their decision making and reporting process. Measurement costs money, but it can also save money through early problem detection and objective clarification of critical software development issues. This report provides some basic concepts that program managers can use to help integrate measurement into the process for managing software development. It also provides an initial set of measures to help address common issues in software intensive acquisitions.

When the Software Acquisition Metrics Working Group first met in 1989, only a few reports existed on the subject of how program managers could use software measurement; now, other reports have been written. The goal of this report is not to compete with those reports, but to use them as starting points for expansion. This report should be viewed not as a standard, but as containing guidelines and advice for program officers and managers starting to use software measurement in their own organizations.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.011.html

CMU/SEI-92-TR-010
ADA254176

Analysis of Reservation-Based Dual-Link Networks

Sha, L.; Sathaye, S.; Strosnider, J.

Next-generation networks are expected to support a wide variety of services. Some services such as video, voice, and plant control traffic have explicit timing requirements on a per-message basis rather than on the average. In this paper, we develop a general model of reservation-based dual-link networks to support real-time communication. We examine the desirable properties of this network and the difficulties in achieving these properties. We then introduce the concept of coherence and develop a theory of coherent dual-link networks. We show that a coherent dual-link network can be analyzed as though it is a centralized system. We then discuss practical considerations in implementing a dual-link network, and implications of this work to address problems observed in the IEEE 802.6 metropolitan area network standard.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.010.html

CMU/SEI-92-TR-009
ADA253327

Parallels in Computer-Aided Design Framework and Software Development Environment Efforts

Dart, S.

This paper is an attempt to raise awareness about the similarities between the efforts of the software development environment (SDE) community and the electronic computer-aided design (CAD) framework community. Apparently, SDE and CAD engineers are not aware of what is happening in each other's fields, yet cross-pollenization of efforts would assist progress. Both communities are addressing the same problems of providing configuration management (CM), tool integration, and process management support in their environment. Each community can benefit from the other since both have similar needs and have found, and are finding, similar solutions. It is particularly useful to consider collaborative efforts as both communities are evolving towards standardization.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.009.html

CMU/SEI-92-TR-008
ADA254175

Past, Present, and Future of Configuration Management, The

Dart, S.

Automated support for configuration management (CM) is one aspect of software engineering environments that has progressed over the last 20 years. The progress is seen by the burgeoning interest in CM, many technical papers and conferences involving CM, a large number of CM tool vendors, and new software development environments that incorporate CM capabilities. This paper is about future issues affecting solutions to CM problems. To put the future into perspective, it is necessary to discuss the past and present situation for CM. The past evolves around CM systems built in-house and supplemented with manual procedures and policies for executing the CM functions. The present consists of a better understanding of CM, the beginnings of a common vocabulary for CM, existence of many third-party CM tools and environments supporting CM, and recognition that a single CM system does not solve all CM problems and that there is a need for better understanding of CM process support. The future involves technical, process-oriented, political, standardization and managerial challenges. These include the need to provide for new CM requirements, understand the effects of advances in environments, deal with governmental requirements on contractors for using certain CM capabilities, and acquire more management commitment for resources in solving the CM problems of an organization. One way to start addressing these challenges is through the definition of a CM services model that provides a conceptual framework for all CM capabilities. As CM is examined more closely in relation to software engineering, it becomes evident that advances in software technology are needed to aid advances in CM technology.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.008.html

CMU/SEI-92-TR-007
ADA253326

Introduction to Software Process Improvement

Humphrey, W. (revised August 1993)

While software now pervades most facets of modern life, its historical problems have not been solved. This report explains why some of these problems have been so difficult for organizations to address and the actions required to address them. It describes the Software Engineering Institute's (SEI) software process maturity model, how this

model can be used to guide software organizations in process improvement, and the various assessment and evaluation methods that use this model. The report concludes with a discussion of improvement experience and some comments on future directions for this work.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.007.html

CMU/SEI-92-TR-006
ADA258234

Proceedings of the CASE Management Workshop

Huff, C.; Smith, D.; Morris, E.; Zarrella, P.

The Software Engineering Institute (SEI) Computer-Aided Software Engineering (CASE) Technology Project sponsored a workshop to address a number of key CASE management issues. The workshop was held at the SEI in Pittsburgh, Pennsylvania on June 19-20, 1991. At the workshop, a representative group of SEI affiliates from industry, government, and academia discussed among themselves such management topics as CASE acquisition policy, what CASE tools can and cannot do, CASE and metrics, and CASE tool selection. The results of these discussions are summarized in this report.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.006.html

CMU/SEI-92-TR-005
ADA253323

Issues and Techniques of CASE Integration with Configuration

Wallnau, K.

Commercial computer-aided software engineering (CASE) tool technology has emerged as an important component of practical software development environments. Issues of CASE tool integration have received heightened attention in recent years, with various commercial products and technical approaches promising to make inroads into this difficult problem. One aspect of CASE integration that has not been adequately addressed is the integration of CASE tools with configuration management (CM)—including both CM policies and systems. Organizations need to address how to make CASE tools from different vendors work effectively with an organization's CM policies and tools (in effect, integrate CASE with CM) within the context of the rapidly evolving state of commercial integration technology. This report describes key issues of the integration of CASE with CM from a third-party integrator's perspective, i.e., how to approach the integration of CASE and CM in such a way as to not require fundamental changes to the implementation of the tools or CM systems themselves.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.005.html

CMU/SEI-92-TR-004
ADA258465

Software Process Development and Enactment: Concepts and Definitions

Feiler, P.; Humphrey, W.

The scientific treatment of the software process is relatively new and, as with any new field, the initial terminology is often confusing. When terms can have a diversity of meanings, technical communication is more difficult and technological progress is constrained. This paper defines a core set of concepts about the software process. These concepts are intended to facilitate communications and to provide a framework for further definitions. The definitions focus on essential concepts; however, they do not represent a comprehensive glossary of common software process terms. Following an initial overview, this paper outlines the basic process concepts which underlie the definitions. The definitions are then grouped in four sets: a framework for process definition, an engineering of process, an enactment of process, and process properties. This is followed by illustrations of the use of these concepts in several domains. The paper concludes with some observations on the definition process.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.004.html

CMU/SEI-92-TR-003
ADA253351

Analysis Technique for Examining Integration in a Project Support Environment, An

Brown, A.; Feiler, P.

In this paper we describe the use of a Project Support Environment (PSE) services reference model as an analysis technique that helps in describing, understanding, and comparing aspects of integration in a PSE. The model is briefly described, before being used as the basis for discussing a number of issues with regard to PSE integration. A major focus of this paper is a discussion of the interfaces of interest in a PSE—interfaces within a single service, between services, and between services and the PSE end-users.

The paper concludes with a discussion of possible interpretations and developments of the model to suit different user requirements.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.003.html

CMU/SEI-92-TR-002
ADA253324

Conceptual Basis for a Project Support Environment Services Model, The

Brown, A.; Feiler, P.

The foundation for a Project Support Environment (PSE) services reference model is presented. This model is to be used as the basis for understanding more about the meaning of integration in a PSE, comparing and contrasting PSE tools and products, and for helping in the identification of PSE interface areas that are candidates for standardization. The model views a PSE as a set of services, distinguishing between services as perceived by PSE end-users, and those provided as mechanisms within the PSE infrastructure. Process constraints on those services are separately identified. The motivation for the view of a PSE is described, followed by a detailed description of the main structure and elements of the model.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.002.html

CMU/SEI-92-TR-001
ADA258325

ADA Validation Tests for Rate Monotonic Scheduling Algorithm

Kohout, K.; Meyer, K.; Goodenough, J.

This report presents a set of tests for checking whether an ADA runtime system properly supports certain rate monotonic scheduling algorithms, specifically, the *basic inheritance* and *priority ceiling* protocols. These tests are intended to be used by vendors and by users to validate implementations of these protocols. The report describes the tests and how they are to be used. The source code is available electronically.

http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.001.html

Special Reports

CMU/SEI-92-SR-013
ADA258259

Software Engineering Process Groups: Results of the 1992 SEPG Workshop Event Evaluation and a First Report on SEPG Status

Miller, M.; Goldenson, D.

This report contains a summary of the results from a questionnaire administered to participants in the Software Engineering Process Group (SEPG) Workshop held in Tysons Corner, Virginia, in April of 1992. The purpose of the questionnaire was twofold. (1) to ask the participants for their judgments about the quality of the week-long event, and (2) to begin collecting information comparing the experiences of existing SEPGs. The participants reported a generally high degree of satisfaction with the content and conduct of the event. Although descriptions about SEPG characteristics and activities apply only to the organizations whose representatives attended the workshop, our findings suggest recent and rapid growth in the SEPG community. Of the 169 responding participants, 72 percent stated that their organizations have SEPGs, and over three-quarters of the SEPGs have been established since 1990.

http://www.sei.cmu.edu/publications/documents/92.reports/92.sr.013.html

CMU/SEI-92-SR-010
ADA2582253

Bibliography of Externally Published Works by the SEI Engineering Techniques Program, A

Brenner, S.; Hart, G.

This bibliography lists works by the members of the Software Engineering Institute (SEI) Engineering Techniques Program that are not published or available from the SEI. The bibliography is organized by type of work (i.e., journal or magazine article, book, or proceedings.) Indices are provided by author and targeted audience.

http://www.sei.cmu.edu/publications/documents/92.reports/92.sr.010.html

CMU/SEI-92-SR-009
ADA257225

Domain-Specific Software Architecture Program, The

Mettala, E. (LTC); Graham, M.

There are six independent projects within the DSSA program. Four of these projects are working in specific, military-significant domains. Those domains are Avionics Navigation, Guidance and Flight Director for Helicopters; Command and Control; Distributed Intelligent Control and Management for Vehicle Management; Intelligent Guidance, Navigation and Control for Missiles. In addition, there are two projects working on underlying support technology. Hybrid (discrete and continuous, non-linear) Control and Prototyping Technology.

This report contains brief descriptions from each project and an overview.

http://www.sei.cmu.edu/publications/documents/92.reports/92.sr.009.html

CMU/SEI-92-SR-008
ADA264798

Annotated Bibliography on Integration in Software Engineering Environments, An

Brown, A.; Penedo, M.

This paper provides an annotated bibliography on integration in software engineering environments (SEEs). The aim is to provide readers with a source of information that can be used as the basis for more detailed studies in this area.

http://www.sei.cmu.edu/publications/documents/92.reports/92.sr.008.html

CMU/SEI-92-SR-004
ADA258255

Reuse-Based Software Development Methodology, A

Kang, K.; Cohen, S.; Holibaugh, R.; Perry, J.; Peterson, A.

Software has been reused in applications development ever since programming started. However, the reuse practices have mostly been ad hoc, and the potential benefits of reuse have never been fully realized. Most of the available software development methodologies do not explicitly identify reuse activities. The Application of Reusable Software Components Project of the Software Engineering Institute is developing a reuse-based software development methodology, and the current direction and the progress of the methodology work are discussed in this paper.

The methodology is based on the life cycle model in DoD-STD-2167A with refinement of each phase to identify reuse activities. The reuse activities that are common across the life cycle phases are identified as: 1) studying the problem and available solutions to the problem and developing a reuse plan or strategy, 2) identifying a solution structure for the problem following the reuse plan, 3) reconfiguring the solution structure to improve reuse at the next phase, 4) acquiring, instantiating, and/or modifying existing reusable components, 5) integrating the reused and any newly developed components into the products for the phase, and 6) evaluating the products. These activities are used as the base model for defining the specific activities at each phase of the life cycle.

This methodology focuses more on identification and application of reusable resources than on construction of reusable resources, and some enhancements in the construction aspect might be necessary to make it more complete.

This methodology has never been applied; it will be used in an application redevelopment experiment and then will be improved based on our experience.

http://www.sei.cmu.edu/publications/documents/92.reports/92.sr.004.html

CMU/SEI-92-SR-003
ADA258468

Joint Integrated Avionics Working Group (JIAWG) Object-Oriented Domain Analysis Method (JODA)

Holibaugh, R.

The Joint Integrated Avionics Working Group (JIAWG) Reuse Subcommittee has initiatives in several areas to demonstrate that reuse can effectively support the JIAWG programs, and the creation of reusable assets is an essential element of reuse. Domain analysis is the process that identifies what is reusable, how it can be structured, and how it can be used. This report describes a method for domain analysis that is based on Coad and Yourdon's "Object Oriented Analysis." This method, the JIAWG Object-Oriented Domain Analysis (JODA), includes several enhancements to the method of Coad and Yourdon and produces a domain model to support asset creation and reuse.

http://www.sei.cmu.edu/publications/documents/92.reports/92.sr.003.html

CMU/SEI-92-SR-001
ADA259854

Report on Senior Executive Seminars on Software Issues

Sisti, F.; Sweet, W.

This report expands on the activities executed by the Software Engineering Institute (SEI) associated with raising the software issue awareness of senior executives in the three principal constituent areas of the SEI: senior defense officials, industry executives, and senior academic personnel. In planning for and executing these activities, the SEI has responded to one of the principal aspects of its charter, which is to address the most important software-related issues applicable to the Department of Defense (DoD) and the software industry in the United States.

http://www.sei.cmu.edu/publications/documents/92.reports/92.sr.001.html

1991 Reports

Technical Reports

CMU/SEI-91-TR-031
ADA248119

Understanding Integration in a Software Development Environment

Brown, A.; Feiler, P.; Wallnau, K.

In the past ten years there has been a great deal of interest in the concept of a Software Development Environment (SDE) as a complete, unifying framework of services supporting most (or all) phases of software development and maintenance. We identify three levels at which the issue of integration in a SDE arises as a key concept at the mechanism level (interoperability of the hardware and basic software), at the end-user services level (combining the methods and paradigms of the various tools), and at the process (adapting end-user services to the working practices of different users, projects and organizations).

In this paper we examine SDEs from an integration perspective, describing the previous work in this area and analyzing the integration issues that must be addressed in an SDE. For illustrative purposes, a particular focus of the paper is the configuration management aspects of an SDE.

Inadequate, incomplete, erroneous, and ambiguous system and software requirements are a major and ongoing source of problems in systems development. These problems manifest themselves in missed schedules, budget excesses, and systems that are to varying degrees unresponsive to the true needs of the sponsor. These difficulties are often attributed to the poorly defined and ill-understood processes used to elicit, specify, analyze, and validate requirements.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.031.html

CMU/SEI-91-TR-030
ADA250415

Requirements Engineering and Analysis Workshop Proceedings

The Software Engineering Institute (SEI) hosted the Requirements Engineering and Analysis Workshop in Pittsburgh, Pennsylvania, on March 12-14, 1991. The intention of the workshop was to focus discussion on issues and activities that could help the Department of Defense (DoD) to deal more effectively with the requirements of mission-critical systems. The SEI workshop built upon work performed previously at the Requirements Engineering and Rapid Prototyping Workshop held by the U.S. Army Communications-Electronics Command (CECOM) Center for Software Engineering in Eatontown, New Jersey, on November 14-16, 1989.

The workshop participants were divided into four working groups: Requirements Engineering Process and Products, Requirements Volatility, Requirements Elicitation, and Requirements Engineering Techniques and Tools. A summary of the findings of each working group follows.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.030.html

CMU/SEI-91-TR-029
ADA244294

Critical Review of the Current State of IPSE Technology, A

Brown, A.

In the past ten years there has been a great deal of interest in the concept of an Integrated Project Support Environment (IPSE) as a complete, unifying framework of services supporting most (or all) phases of software development and maintenance. In this paper we evaluate the current state of research work in this area, suggest some reasons for the relative lack of success, and make proposals for ensuring measured progress in the future.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.029.html

CMU/SEI-91-TR-028
ADA256590

Application of Feature-Oriented Domain Analysis to the Army Movement Control Domain and Appendices A-I

Cohen, S.; Stanley Jr., J.; Peterson, A.; Krut Jr., R.

This report documents an analysis of the army movement control domain performed by the Software Engineering Institute (SEI) and a team of movement control experts from the Army. This report includes common terminology and requirements extracted from Army doctrine, experts in the field, and movement control systems. The report also describes the potential for prototyping of systems using domain analysis products and the tool support needed.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.028.html

CMU/SEI-91-TR-027
ADA245051

Comparison of U.S. and Japanese Software Process Maturity, A

Humphrey, W.; Kitson, D.; Gale, J.

This report characterizes the software processes currently used by software managers and practitioners in the U.S., Japan.

The U.S. data for this comparative study of the state of software practice in the U.S., Japan is drawn from extensive SEI assessments conducted from 1987 through 1989. The Japanese maturity data was gathered during a three-week trip to Japan by the authors. This includes data on 168 U.S. projects and 196 Japanese software projects.

This data is not a statistically valid sample of the software capabilities of either the U.S. or Japanese software industries. However, the data does indicate that the Japanese software industry is, in some respects, both weaker and stronger than that in the U.S. There are also significant differences in the maturity findings between the U.S., Japanese organizations, and these have important implications for software organizations in both countries.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.027.html

CMU/SEI-91-TR-026
ADA242129

Application-Level Implementation of the Sporadic Server, An

González Harbour, M.; Sha, L.

The purpose of this paper is to introduce a sporadic server algorithm that can be implemented as an application-level task, and that can be used when no runtime or operating system level implementation of the sporadic server is available. The sporadic server is a simple mechanism that both limits and guarantees a certain amount of execution power dedicated to servicing aperiodic requests with soft or hard deadlines in a hard real-time system. The sporadic server is event-driven from an application viewpoint, but appears as a periodic task for the purpose of analysis and, consequently, allows the use of analysis methods such as rate monotonic analysis to predict the behavior of the real-time system.

When the sporadic server is implemented at the application-level, without modification to the runtime executive or the operating system, some of its requirements cannot be met strictly and, therefore, some simplifications need to be assumed. We show that even with these simplifications, the application-level sporadic server proposed in this paper has the same worst-case performance as the full-featured runtime sporadic server algorithm, although the average case performance is slightly worse. The implementation requirements are a runtime prioritized preemptive scheduler and system calls to change a task's or thread's priority. Two implementations are introduced in this paper, one ADA and the other for POSIX 1003.4a, Threads Extension to Portable Operating Systems.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.026.html

CMU/SEI-91-TR-025
ADA240604

Key Practices of the Capability Maturity Model

Weber, C.; Paulk, M.; Wise, C.; Withey, J.

This document, *Key Practices of the Capability Maturity Model*, provides the key practices that correspond to the key process areas at each maturity level of the Capability Maturity Model and information on how to interpret the key practices.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.025.html

CMU/SEI-91-TR-024
ADA240603

Capability Maturity Model for Software

Paulk, M.; Curtis, B.; Averill, E.; Bamberger, J. Kasse, T.; Konrad, M.; Perdue, J., Weber, C.; Withey, J.

In November 1986, the Software Engineering Institute with assistance from the Mitre began developing a process maturity framework that would assist organizations in improving their software process. This effort was initiated in response to a request to provide the federal government with a method for assessing the capability of their software contractors. in September 1987, the SEI released a brief description of the process maturity framework and a maturity questionnaire (CMU/SEI-87-TR-023). The SEI intended the maturity questionnaire to provide a simple tool for identifying areas where an organization's software process needed improvement. Unfortunately, the questionnaire was too often regarded as "the model" rather than as a vehicle for exploring process maturity issues.

After four years of experience with the software process maturity framework and the preliminary version of the maturity questionnaire, the SEI has evolved the software process maturity framework into a fully defined model. This model will be used in a systematic, principled way to derive a maturity questionnaire. By fully elaborating the maturity framework, a model has emerged that provides organizations with more effective guidance for establishing process improvement programs that were offered by the maturity questionnaire. Using knowledge acquired from software process assessments and extensive feedback from both industry and government, an improved version of the process maturity framework and been produced called the Capability Maturity Model for Software (CMM). This paper is an introduction to the revised model.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.024.html

CMU/SEI-91-TR-022
ADA244697

Building Distributed ADA Applications from Specifications and Functional Components

Doubleday, D.; Barbacci, M.; Weinstock, C.; Gardner, M.; Lichota, R.

Durra is a language and support environment for the specification and execution of distributed ADA applications. A Durra programmer describes an application as a collection of processes and data links. More complicated application descriptions may also include a structuring of this collection that varies dynamically according to a set of reconsideration conditions. Each process defined in the application description is associated with an independently compiled ADA subprogram that implements the behavior of that process. The Durra programmer specifies the distribution of application components by assigning them to virtual nodes called clusters. For each cluster, the Durra compiler generates a multithreaded ADA program that imports the code for the processes assigned to that node and manages their execution. Durra also facilitates rapid prototyping through the use of tools that interpret timing specifications associated with processes and generates ADA code to simulate their expected behavior.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.022.html

CMU/SEI-91-TR-021
ADA242128

Durra: An Integrated Approach to Software Specification, Modeling, and Rapid Prototyping

Barbacci, M.; Doubleday, D.; Weinstock, C.; Lichota, R.

Software specification, modeling, and prototyping activities are often performed at different stages in a software development project by individuals who use different specialized notations. The need to manually interpret and transform information passed between stages can significantly decrease productivity and can serve as a potential source of error. Durra is a non-procedural language designed to support the development of distributed applications consisting of multiple, concurrent, large-grained tasks executing in a heterogeneous network. Durra provides a framework through which one can specify the structure of an application in conjunction with its behavior, timing, and implementation dependencies. These specifications may be validated by passing behavioral and timing information associated with each Durra task description to a run-time interpreter. Similarly, software prototypes may be constructed by directing this information to a suitable source code generator. We have already developed an interpreter and source code translator for a language based on simple timing expressions. We are presently constructing a source code generator for a more complex language defined by SMARTS (the Specification Methodology for ADAptive Real-Time systems developed by Hughes Aircraft Company).

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.021.html

CMU/SEI-91-TR-020
ADA245051

Design Specifications for ADAptive Real-Time Systems

Lichota, R.

The design specification method described in this report treats a software architecture as a set of run-time entities, including tasks and external input/output elements, which interact either via messages or shared data structures. Tasks have a single thread of execution and represent program units that may be executed concurrently. External input elements produce input requests which in turn trigger a set of low level activities to be executed by tasks. External output elements consume results which are produced by tasks. The specification method discussed here facilitates the description of the dynamic structure of run-time entities, the synchronization and communication between these entities, and their resource consumption and production properties (which include timing and sizing).

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.020.html

CMU/SEI-91-TR-019
ADA248118

Description of Cluster Code Generated by the Durra Compiler, A

Doubleday, D.; Gardner, M.; Weinstock, C.

Durra is a language and support environment for the specification and execution of distributed ADA applications. The Durra programmer specifies the distribution of application components by assigning them to virtual nodes called clusters. For each cluster named in an application description, the Durra compiler generates an ADA package body with a standardized format. Within the confines of the format, the content of the package body varies according to the requirements placed upon the cluster by the Durra application description. The cluster-specific package body is compiled and linked with a fixed set of ADA compilation units, common to all clusters, to form a multitasking ADA program. The intended audience for this document is Durra application developers, who will need an understanding of the concepts presented here in order to be effective Durra application debuggers.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.019.html

CMU/SEI-91-TR-018
ADA246405

Durra: A Task-Level Description Language Reference Manual (Version 3)

Barbacci, M.; Doubleday, D.; Gardner, M.; Lichota, R.; Weinstock, C.

Durra is a language designed to support the development of distributed programming applications consisting of concurrent, large-grained processes devoted to specific pieces of the application. During execution time the application processes run on possibly separate processors, and communicate with each other by sending messages of different types across communication links. The application developer is responsible for prescribing a way to manage all of these resources, called a task-level application description. It describes the processes to be executed, the assignments of processes to processors, and the communication channels required to transmit messages data between processes. Durra is a task-level description language, a notation in which to write these application descriptions.

This document is a revised version of the original reference manual. It describes the syntax and semantics of the language and incorporates all the language changes introduced as a result of our experiences writing application descriptions in Durra.

A companion document, *Durra: A Task-Level Description Language User's Manual*, describes how to use the compiler and support tools.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.018.html

CMU/SEI-91-TR-017
ADA240712

Issues in Real-Time Data Management

Graham, M.

This report explores issues related to the use of database management technology in support of real-time systems programming. It describes the potential benefits of database support for real-time systems, and it describes the state of the art in database technologies relevant to real-time. The report concludes that more research and development will be needed before the benefits of database management can be applied to real-time system development.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.017.html

CMU/SEI-91-TR-016
ADA241781

Measurement in Practice

Rifkin, S.; Cox, C.

A few organizations have reputations for implementing excellent software measurement practices. A sample of these organizations was surveyed in site visits. Clear patterns of practices emerged and they are reported at a consolidated, "lessons learned" level and in more detailed case studies.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.016.html

CMU/SEI-91-TR-014
ADA253362

Proceedings of the CASE Adoption Workshop

Huff, C.; Smith, D.; Stepien-Oakes, K.; Morris, E.; Zarrella, P.

The Software Engineering Institute (SEI) CASE Technology Project sponsored a workshop to address a number of key CASE adoption issues. The workshop was held at the SEI in Pittsburgh, Pennsylvania on November 13-14, 1990. At the workshop, a representative group of SEI affiliates from industry, government, and academia discussed among themselves such adoption topics as CASE benefits, realistic CASE budget estimates, CASE tool fit, CASE adoption roles, and factors in the project success of CASE. The results of these discussions are summarized in this report.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.014.html

CMU/SEI-91-TR-013
ADA248152

CASE Studies in Environment Integration

Morris, E.

Four environment builders and participants at two workshops were queried concerning the environment standards, implementations, and technology that prove useful in the integration of tools into software engineering environments. Specific information was gathered about the software and hardware environments in which tool integration occurred, the goals of integration, the tools integrated, mechanisms used, and the standards applied. Observations concerning the current state of tool and environment integration are provided, and trends in integration are identified.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.013.html

CMU/SEI-91-TR-012
ADA240851

Notes on Applications of the SQL ADA Module Description Language (SAMeDL)

Chastek, G.; Graham, M.; Zelesnik, G.

The SQL ADA Module Description Language (SAMeDL) is a language for describing information services to be provided to ADA application programs by SQL database management systems. This report shows how the SAMeDL can be adapted and extended to provide services to applications needing advanced features (e.g., dynamic SQL, or using non-ANSI standard data types (decimal, date) or having other unusual requirements. It also contains short descriptions of some implementation details.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.012.html

CMU/SEI-91-TR-011
ADA237810

Tool Integration and Environment Architectures

Wallnau, K.; Feiler, P.

The expanding CASE market is having substantial impact on software development environment technology in the area of environment support for tool integration. Sharpened awareness of CASE integration requirements, particularly in the context of the larger number of fully developed CASE tools, has resulted in a technology shift away from monolithic integrated project support environments (IPSE) derived form the Stoneman model in favor of highly distributed environments based upon a federation of environment services. Federated environments promise environment framework support for the reuse of a large number of existing CASE tools and the development of highly interactive, tightly-integrated CASE environments. The evolution of environment framework technology to

support CASE federation is predicated on an improved understanding of the techniques and issues of tool integration. One reflection of this improved understanding is recognition of the need to address integration mechanisms, tool semantic integration, and tool process integration as separate but related issues. This paper describes the evolution of environment architectures to support federated CASE integration and outlines the implications of this evolution on the technical issues of CASE tool integration.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.011.html

CMU/SEI-91-TR-010
ADA241780

Models for Undergraduate Project Courses in Software Engineering

Shaw, M.; Tomayko, J.

The software engineering course provides undergraduates with an opportunity to learn something about real-world software development. Since software engineering is far from being a mature engineering discipline, it is not possible to define a completely satisfactory syllabus. Content with a sound basis is in short supply, and the material most often taught is at high risk of becoming obsolete within a few years.

Undergraduate software engineering courses are now offered in more than a hundred universities. Although three textbooks dominate the market, there is not yet consensus on the scope and form of the course. The two major decisions an instructor faces are the balance between technical and management topics and the relation between the lecture and project components. We discuss these two decisions, with support from sample syllabi and survey data on course offerings in the United States and Canada. We also offer some advice on the management of a project-oriented course.

This report will also appear as Carnegie Mellon University, School of Computer Science Technical Report No. CMU-CS-91-174.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.010.html

CMU/SEI-91-TR-009
ADA250039

Software Engineering Education Directory

This directory provides information about software engineering courses and software engineering degree programs offered by colleges and universities, primarily in the United States.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.009.html

CMU/SEI-91-TR-008
ADA244292

Issues in Tool Acquisition

Zarrella, P.; Smith, D.; Morris, E.

This technical report identifies issues involved in the acquisition of Computer Aided Software Engineering (CASE) tools. Among the issues identified and discussed are cost, performance, process support, maintenance, data management, tool integration, and standardization. The report concludes with recommendations intended for individuals or groups responsible for acquiring CASE tools.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.008.html

CMU/SEI-91-TR-007
ADA235782

Configuration Management Models in Commercial Environment

Feiler, P.

A number of advances can be observed in recent commercial software development environments in support of configuration management (CM). These advances include: configurations as managed objects; transparent access to repositories; and, in addition to the familiar checkout/checkin model, three CM models based on configurations. These CM models are the composition model, the long transaction model, and the change set model. Typically, one or two of the models can be found in a system.

This report analyzes the models with respect to their potential impact on the software development process, resulting in several observations. Some of the models exist in a number of variations, each impacting the software process differently. CM capabilities can be found not only in CM tools and environment frameworks, but also in development tools. Integration of such tools into environment raises the need for different CM models to interoperate. Therefore, it is desirable to evolve to a unified CM model that encompasses the full range of CM concepts and can be adapted to different process needs.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.007.html

CMU/SEI-91-TR-006
ADA235641

Rate Monotonic Analysis for Real-Time Systems

Sha, L.; Klein, M.; Goodenough, J.

The essential goal of the Rate Monotonic Analysis (RMA) for Real-Time Systems Project at the Software Engineering Institute is to catalyze improvement in the practice of real-time systems engineering, specifically by increasing the use of rate monotonic analysis and scheduling algorithms. In this report, we review important decisions in the development of RMA. Our experience indicates that technology transition considerations should be embedded in the process of technology development from the start, rather than as an afterthought.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.006.html

CMU/SEI-91-TR-005
ADA244293

Evaluation of Process Modeling Improvements

Krut Jr., R.; Wood, D.

The SEI has been involved with the development and analysis of software process models for several years. As part of the ongoing process of technology evolution, a study has been undertaken to experimentally implement a set of proposed improvements to the process modeling techniques used by the SEI, and to evaluate the results of that experimentation. As a result of that study, a number of modifications to our techniques have been identified. These modifications enhance the support of software engineering concepts in the development and use of process models. This report describes the study and elaborates upon the advantages and disadvantages of the proposed technique improvements.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.005.html

CMU/SEI-91-TR-004
ADA235780

Rationale for SQL ADA Module Description Language SAMeDL

Chastek, G.; Graham, M.; Zelesnik, G.

The SQL ADA Module Description Language, SAMeDL, is a language for the specification of Abstract Interfaces as delineated by the SQL ADA Module Extensions (SAME) methodology. The language is formally defined in the SAMeDL Reference Manual. This document is a companion to the Reference Manual. Whereas the Reference Manual is meant to be precise, the Rationale is meant to be clear.

An explanation of the problem solved by the SAMeDL is given. The creation of a new language is justified and the underlying principles of that language are described. Crucial issues in the language are then explained. These include:

- The form and meaning of identifiers in the SAMeDL.
- The role of and procedures for data definition in the SAMeDL. This includes support for enumerations and constants in the SAMeDL.
- The typing rules of the SAMeDL.

The proposed use of some SAMeDL features is also illustrated. These include Standard Post Processing and User Defined Base Domains.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.004.html

CMU/SEI-91-TR-002
ADA236340

1991 SEI Report on Graduate Software Engineering Education

Ford, G.

This report on graduate software engineering education presents a variety of information for university educators interested in establishing a software engineering program. This includes a model curriculum, an annotated bibliography of software engineering textbooks, and descriptions of major software engineering research journals. It also includes detailed descriptions of the SEI Academic Series of videotape courses, which constitute an example of an implementation of the model curriculum. Twenty-three university graduate programs in software engineering are surveyed. Software engineering textbooks and research journals are also surveyed.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.002.html

CMU/SEI-91-TR-001
ADA235698

Formal Development of ADA Programs Using Z and Anna: A Case Study

Place, P.; Wood, W.

This report describes a method for the formal development of ADA programs from a formal specification written in Z. ANNotated ADA (Anna) is used as an intermediate language linking the more abstract Z specifications to the concrete ADA program. The method relies on the notion that successive small transformations of a specification are easier to verify than a few large transformations. Essentially the method uses three notations for the representation of the system: an implementation-independent notation for the specification of the system, an implementation-dependent notation for the representation of a lower level specification of the system, and the implementation language. Z and Anna are outlined briefly and examples of transformations are presented. A simple Z specification has been chosen and the transformations presented in the report are transformations of the Z specification into Anna. Conclusions are drawn about the development method presented.

http://www.sei.cmu.edu/publications/documents/91.reports/91.tr.001.html

Special Reports

CMU/SEI-91-SR-013
ADA255376

Fault Tolerant Systems Practitioner's Workshop June 10-11, 1991

Heimerdinger, W.; Weinstock, C.

On June 10-11, 1991, a Fault Tolerant Systems Practitioner's Workshop was held at the Software Engineering Institute. The purpose of the workshop was to attempt to identify how fault tolerance is being applied today, why fault tolerance is under used, and what can be done to bring fault tolerant practices into wider use. Attendance at the workshop was limited to a small number of practitioners who had successfully applied fault tolerance in a systems context. This report summarizes the proceedings of the workshop which included a discussion of barriers to the deployment of fault tolerant systems, a summary of the state of the practice, and a discussion of the technology needs of fault tolerance. The report concludes with a discussion of ways the Software Engineering Institute may be able to help bring fault tolerant practices into wider use.

http://www.sei.cmu.edu/publications/documents/91.reports/91.sr.013.html

CMU/SEI-91-SR-004
ADA2523646

Comparison of ADA 83 and C++, A

Weiderman, N.

The purpose of this report is to provide technical input to the Deputy Assistant Secretary of the Air Force for Communications, Computers, and Logistics to assist that office in preparing a business case for using ADA or C++ to develop Corporate Information Management (CIM) systems. This technical input has been gathered by using the comparison methodology of a 1985 Federal Aviation Administration (FAA) report as a model, as well as by conducting interviews with experts in ADA and C++. The conclusion of this report is that technically neither language is clearly better than the other; for government use, however, there is clear justification and rationale for using ADA rather than C++ for large complex systems with long lifetimes.

http://www.sei.cmu.edu/publications/documents/91.reports/91.sr.004.html

CMU/SEI-91-SR-003
ADA248117

Context Analysis of the Movement Control Domain for the Army Tactical Command and Control System (ATCCS), A

Peterson, A.; Cohen, S.

This report describes the results of the first phase of a domain analysis performed by the SEI's Domain Analysis Project. The report establishes the context and scope of the *movement control* domain within command and control (C^2) systems and in particular, ATCCS. It discusses the purpose of a domain analysis and presents an overview of the method being used to perform the analysis. The report describes both the Army's use of ATCCS in achieving the Army mission and the ATCCS software development strategy. A brief description of Army C^2systems lays a foundation for defining movement control and its context in the ATCCS structure. Within this context, the report identifies the key areas within movement control that will be explored during the remainder of the domain analysis. The report also describes the approach for performing the analysis and includes appendices providing additional information about ATCCS, relevant terms, and acronyms.

http://www.sei.cmu.edu/publications/documents/91.reports/91.sr.003.html

Educational Materials

CMU/SEI-91-EM-006
ADA242547

Materials to Support Teaching a Project-Intensive Introduction to Software Engineering

Tomayko, J.

In a project-intensive introduction to software engineering at Carnegie Mellon University (CMU), students developed an interactive exhibit for the Kansas Cosmosphere and Space Center. In a similar course at The Wichita State University, students modified the original CMU project to fit new customer requirements. This package of educational materials provides the software products and documents developed by the students in these courses, along with lesson plans, exams, and other materials used by the instructors.

http://www.sei.cmu.edu/publications/documents/ems/91.em.006.html

CMU/SEI-91-EM-005
ADA2406710

Scenes of Software Inspections: Video Dramatizations for the Classroom

Deimel, L.

This report describes the videotape *Scenes of Software Inspections*, which contains brief dramatizations that demonstrate appropriate and inappropriate conduct of software inspections. The tape also includes scenes that show other kinds of group interactions. Any of these scenes can be incorporated into lectures, self-study materials, or other educational delivery mechanisms, to illustrate how to perform inspections, an important software engineering technique.

http://www.sei.cmu.edu/publications/documents/ems/91.em.005.html

CMU/SEI-91-EM-004
ADA242548

Software Engineering Project Course with a Real Client, A

Bruegge, B.; Cheng, J.; Shaw, M

At Carnegie Mellon University, we taught an introductory software engineering course that was organized around a project with a real deliverable for a real client. This case study describes the background and organization of the course and presents the lecture and project materials produced by the faculty and students of the course.

http://www.sei.cmu.edu/publications/documents/ems/91.em.004.html

1990 Reports

Technical Reports

CMU/SEI-90-TR-032
ADA235776

STARS/Users Workshop: Final Report—Issues for Discussion Groups

Bamberger, J. (ed.)

The STARS (Software Technology for ADAptable, Reliable Systems) Program is focused on providing the DoD software community with a software engineering environment, repository technology, and process models. This STARS Workshop was targeted toward increasing the communication between the STARS Program and the builders of software-dependent systems. This was the first of many public discussion hosted by the STARS Program. This workshop was hosted by the SEI.

CMU/SEI-90-TR-026
ADA235781

SQL ADA Module Description Language SAMeDL Version 3.75, The

Graham, M.

This document is a reference manual for the SQL ADA Module Description Language (SAMeDL). The SAMeDL is used to describe database services needed by ADA application programs.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.026.html

CMU/SEI-90-TR-025
ADA235639

Tool Version Management Technology: A Case Study

Feiler, P.; Downey, G.

This report describes a portion of the problem of maintaining tools for the purpose of software development. It discusses an innovative solution to tool version management available in the commercially available Network Software Environment from Sun Microsystems, Inc. It applies NSE mechanisms to solve three problems that are common to the use and management of tools for software development.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.025.html

CMU/SEI-90-TR-024
ADA235784

Software Engineering Process Group Guide

Fowler, P.; Rifkin, S.

Improving the of software systems development and maintenance is the most reliable way to improve product quality. This document offers guidance on how to establish a software engineering process group (SEPG) and related software engineering process improvement functions. The process group works with line organizations to improve process quality by helping to assess current status, plan and implement improvements, and transfer technology to facilitate improvement in practice.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.024.html

CMU/SEI-90-TR-023
ADA235510

Transaction-Oriented Configuration Management: A Case Study

Feiler, P.; Downey, G.

Software configuration management (SCM) is a key element of the software development process. A number of new configuration management techniques in commercial SCM tools and environments with SCM capabilities have been observed. This report illustrates some of the advances in SCM concepts by example of a particular commercial system: the Sun Network Software Environment (NSE). NSE embodies a transaction model of configuration management. In order to demonstrate the capabilities and limitations of the transaction model, NSE is applied to three problem areas for configuration management: adaptation for parallel development and team support, development and maintenance in software families and development in a distributed and heterogeneous network.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.023.html

CMU/SEI-90-TR-022
ADA237049

Design Space and Design Rules for User Interface Software Architecture, A

Lane, T.

The architecture of a user interface software system can be described in terms of a fairly small number of key functional and structural choices. This report presents a "design space" that identifies these key choices and classifies the alternatives available for each choice. The design space is a useful framework for organizing and applying design knowledge. The report presents a set of design rules expressed in the terms of the design space. These rules can help a software designer to make good structural choices based on the functional requirements for a user interface system. Extension of this work might eventually provide automated assistance for structural design.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.022.html

CMU/SEI-90-TR-021
ADA235785

Feature-Oriented Domain Analysis (FODA) Feasibility Study

Kang, K.; Cohen, S.; Hess, J.; Novak, W.; Peterson, A.

Successful software reuse requires the systematic discovery and exploitation of commonality across related software systems. By examining related software systems and the underlying theory of the class of systems they represent, domain analysis can provide a generic description of the requirements of that class of systems and a set of approaches for their implementation. This report will establish methods for performing a domain analysis and describe the products of the domain analysis process. To illustrate the application of domain analysis to a representative class of software systems, this report will provide a domain analysis of window management system software.

CMU/SEI-90-TR-020
ADA235751

Prospects for an Engineering Discipline of Software

Shaw, M.

Software Engineering is not yet a true engineering discipline, but it has the potential to become one. Older engineering fields offer glimpses of the character software engineering might have. From these hints and an assessment of the current state of software practice, we can project some characteristics software engineering will have and suggest some steps toward an engineering discipline of software.

This paper begins by examining the usual practice of engineering and the way it has evolved in other disciplines. This discussion provides a historical context for assessing the current practice of software production and setting out an agenda for attaining an engineering practice.

CMU/SEI-90-TR-019
ADA226724

Analysis of Input/Output Paradigms for Real-Time Systems, An

Klein, M.; Ralya, T.

The correctness of a real-time system with hard deadline requirements depends both on the logical correctness and on the timing correctness of the system. The principles of rate monotonic scheduling have proven to be very useful in providing a framework for designing, analyzing, and modifying the timing and concurrency aspects of real-time

systems. This paper illustrates how to build a mathematical model of the schedulability of a real-time system, taking into consideration such factors as preemption, synchronization, non-preemptibility, interrupts, and process idle time. In particular, this paper illustrates how these principles can be applied to input/output interfaces (e.g., to devices or local area networks) to predict the timing behavior of various design alternatives.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.019.html

CMU/SEI-90-TR-018
ADA235737

Studying Software Architecture Through Design Spaces and Rules

Lane, T.G

This report argues that the overall structure of software systems ("software architecture") is usefully studied by constructing design spaces. A design space identifies the key functional and structural choices made in creating a system design, and it classifies the alternatives available for each choice. Rules can be formulated to relate choices within a design space. Sets of such rules are a valuable design aid and offer a promising route to automatic structural design. By codifying design practice, design spaces can also aid software maintenance and training. To support this argument, the report describes a design space and associated rules for user interface software, and it discusses an experiment that validated these design rules by comparing their predictions to real system designs.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.018.html

CMU/SEI-90-TR-017
ADA226693

Experiences Porting the Distributed ADA Real-Time Kernel

Tomayko, J.; Smith, B.

The Distributed ADA Real-Time Kernel (DARK) is a mechanism for supporting the execution of distributed real-time ADA applications in embedded computer systems. It provides a solution to scheduling and distributing tasks without modifying the ADA language or vendor-supplied run time systems. An important test of the utility of the Kernel is whether or not it can be ported to different hardware architectures and still function effectively. As part of an independent research and development project, Boeing Military Airplanes and The Wichita State University became co-acceptors of a copy of DARK for the purpose of demonstrating a port to a 68000-based distributed architecture. This technical report describes the experiences in accomplishing the port.

CMU/SEI-90-TR-015
ADA235783

Informatics for a New Century: Computing Education for 1990s and Beyond

Shaw, M.

Information technology and computer science have not only reshaped computation, communication, and commerce; they have expanded the basic models and paradigms of many disciplines. Informatics education has obligations to all the communities that rely on information technology, not just the computing professionals. Serving this extended audience well requires changes in the content and presentation of computing curricula. This paper sketches the coming needs for information processing and analyzes the populations that will require informatics education. It considers curriculum requirements through two examples, one outside the traditional boundary of computer science and one inside.

CMU/SEI-90-TR-014
ADA235640

CASE Tool Integration and Standardization

Zarrella, P.

CASE tool users are faced with the task of coordinating tools and data from a variety of sources spanning the entire software development life cycle. Despite much discussion and increased standardization activity, complete, transparent CASE tool integration is still a long way from realization. There are a number of factors which have complicated the tool integration scenario and a number of actions being taken in an attempt to resolve the problems. The implications of these concerns can be examined from the perspectives of single-vendor, multiple-vendor, operating environment, development process, and end-user integration. In addition to specific technical and

methodological solutions, standards efforts are viewed as a possible path to tool integration. To date, formal efforts have done little to resolve the integration problems, but de facto standards may well become the cornerstone of future CASE tool evolution.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.014.html

CMU/SEI-90-TR-012
ADA226694

National Software Capacity: Near-Term Study

Siegel, J.; Stewman, S.; Konda, S.; Larkey, P.; Wagner, W.

This study provides an initial assessment of the U.S.'s industrial capacity to produce MCCR software. A survey of senior government and industry people showed that 90 percent of them expected a serious problem with the nation's capacity to produce military software over the next 5 years. They ranked acquisition and labor factors as contributing most to the failure of military system development contracts to meet schedule or costs. The study team also analyzed available data about the supply of labor (new graduates and experienced scientists and engineers) and three aspects of demand (ADA systems, PDSS, and related commercial applications) before concluding there is a serious capacity problem. The report describes labor, organizational, and technological issues affecting software production capacity and concludes with some preliminary recommendations for DoD and industry initiatives.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.012.html

CMU/SEI-90-TR-011
ADA235753

Spectrum of Functionality in Configuration Management Systems

Dart, S.

The Software Environments Project at the Software Engineering Institute has found considerable progress concerning support for software configuration management (CM) in environments and tools. This paper's intent is to highlight a spectrum of features provided by existing CM systems. The spectrum shows features as being extensions or generalizations of other features and these extensions represent the progress. As part of presenting the features, the scope of issues concerning users of CM systems is discussed. No single CM system provides all the functionality required by the different kinds of users of CM systems. Rather, each CM system addresses some part of a spectrum of functionality. To complete the report, several configuration management systems are briefly described.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.011.html

CMU/SEI-90-TR-008
ADA235752

Generic Avionics Software Specification

Locke, D.; Goodenough, J.

This report informally specifies the general functions, data interactions, and timing constraints for an avionics mission control computer system typical of those found in some existing U.S. Navy/Marine Corps aircraft. Avionics functions and equipment are described only to the extent needed to specify MCC behavior, with primary emphasis being given to timing constraints and data interactions. The specification is intended to allow a wide variety of possible implementation approaches and was developed primarily to exemplify timing requirements and functional interactions in a typical real-time system.

The specification consists of an introductory description of the environment in which the mission computer operates followed by a functional description of the requirements imposed on the mission computer. The functional description is minimal and serves mainly to characterize computer workload and data interactions. An appendix contains a summary of the timing requirements together with an analysis of the expected CPU load for a particular attack scenario.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.008.html

CMU/SEI-90-TR-007
ADA226817

Hartstone Benchmark Results and Analysis

Donohoe, P.

Hartstone is a series of timing requirements for testing a system's ability to handle hard real-time applications. It is specified as a set of processes with well-defined workloads and timing constraints. The name Hartstone derives from *HA*rd *R*eal *T*ime and the fact that the workloads are based on the well-known Whetstone benchmark. This report describes the results obtained by running Version 1.0 of the Hartstone benchmark, an ADA implementation of one of the requirements, on a number of compiler/target processor combinations. The characteristics and expected behavior of the benchmark are described, actual results are presented and analyzed, and the lessons learned about the compilers and processors, and the benchmark itself, are discussed. Nothing in this report should be taken as an endorsement of, or an indictment of, a particular product. Users of ADA technology are encouraged to experiment with the Hartstone benchmark relative to their own particular application requirements.

http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.007.html

CMU/SEI-90-TR-006
ADA226723

Implementing Sporadic Servers in ADA

Sprunt, B.; Sha, L.

The purpose of this paper is to present the data structures and algorithms for implementing sporadic servers in real-time systems programmed in ADA. The sporadic server algorithm is an extension of the rate monotonic scheduling algorithm. Sporadic servers are tasks created to provide limited and usually high-priority service for other tasks, especially aperiodic tasks. Sporadic servers can be used to guarantee deadlines for hard-deadline aperiodic tasks and provide substantial improvements in average response times for soft-deadline aperiodic tasks over polling techniques. Sporadic servers also provide a mechanism for implementing the Period Transformation technique that can guarantee that a critical set of periodic tasks will always meet their deadlines during a transient overload. Sporadic servers can also aid fault detection and containment in a real-time system by limiting the maximum execution time consumed by a task and detecting attempts to exceed a specified limit. This paper discusses two types of implementations for the sporadic server algorithm: (1) a partial implementation using an ADA task that requires no modifications to the ADA runtime system and (2) a full implementation within the ADA runtime system. The overhead due to the runtime sporadic server implementation and options for reducing this overhead are discussed. The interaction of sporadic servers and the priority ceiling protocol is also defined.

CMU/SEI-90-TR-005
ADA223741

Survey of Formal Specification Techniques for Reactive Systems

Place, P.; Wood, W.

Formal methods are being considered for the description of many systems including systems with real-time constraints and multiple concurrently executing processes. This report develops a set of evaluation criteria and evaluates Communicating Sequential Processes (CSP), the Vienna Development Method (VDM), and temporal logic. The evaluation is based on specifications, written with each of the techniques, of an example avionics system.

CMU/SEI-90-TR-003
ADA223881

1990 SEI Report on Undergraduate Software Engineering Education

Ford, G.

Fundamental issues of software engineering education are presented and discussed in the context of undergraduate programs. Included are discussions of the definition of software engineering and its differences from computer science, the need for undergraduate software engineering education, possible accreditation of undergraduate programs, and prospects for professional certification and licensing of software engineers. The objectives and content of an undergraduate program are described, as are strategies for the evolution and implementation of such programs. An appendix presents a report on the 1989 SEI Workshop on an Undergraduate Software Engineering Curriculum.

Special Reports

CMU/SEI-90-SR-012
ADA227564

National Software Capacity: Near-Term Study (Executive Summary)

Siegel, J.; Stewman, S.; Konda, S.; Larkey, P.; Wagner, W. G.

This document is a summary of the results of the near-term study. The complete results are published in CMU/SEI-90-TR-012.

http://www.sei.cmu.edu/publications/documents/90.reports/90.sr.012.html

CMU/SEI-90-SR-010
ADA226725

Understanding the Adoption of ADA: Results of an Industry Survey

Carlson, M.; Smith, G.

In 1983, the U.S. Department of Defense (DoD) established a policy requiring the use of a new programming language, ADA, for the development of all new Mission-Critical Computer Resource (MCCR) software that it purchases. Firms that supply the DoD with these systems have shown considerable variation in their decisions to incorporate this new technology into their products and production processes. This survey is part of a multi-stage research project that sought to understand the variability in firms' adoption and use of new information technologies. The present report is a follow-up and elaboration on a case study of the adoption of ADA which is described in CMU/SEI-89-TR-028, *Understanding the Adoption of ADA: A Field Study Report*.

Participants in the survey were 123 business and technical people from 69 business units that supply the DoD with MCCR software systems and services. The survey explored factors pertaining to respondents' technical and market environments in an attempt to describe depth of adoption and to describe the differences between the firms with active ADA contracts and those without active contracts. For firms that have adopted ADA the report describes aspects of the language and tools that are considered most useful in different application areas. At present, 85% of the units have proposed to use ADA as a primary implementation language, and 70% have been awarded a contract in which ADA is the primary implementation language. Within the context of this study, ADA contract awards have been in the following application areas: aircraft engines, attack radar, display processors, flight control, flight trainers, ground control vehicles, night vision, radar warning receivers, missiles, space command and control, and tactical command and control. Survey participants reported that ADA is being used in 50% of the new development contracts and is being proposed for use in 60% of the contracts in the proposal stage.

http://www.sei.cmu.edu/publications/documents/90.reports/90.sr.010.html

CMU/SEI-90-SR-006
ADA226695

DARK Technology Transition Plan

Bamberger, J.

The DARK Transition Project Plan is aimed at the transition of the concepts, models, and prototype implementation developed by the DARK development project over the past two years. This document presents the background, rationale, and conceptual goals of the DARK Project.

1. Provides an overview of the phase approach to Technology Transition used within the SEI and its relationship to DARK Project activities.
2. Describes the goals and objectives of the DARK Transition Project in the context of the phase approach.
3. Defines the tasks constituting the DARK Transition Project.

http://www.sei.cmu.edu/publications/documents/90.reports/90.sr.006.html

CMU/SEI-90-SR-003
ADA248089

Domain Analysis Bibliography, A

Hess, J.; Cohen, S.; Holibaugh, B.; Kyang, K.; Peterson, A.; Novak, W.; Carroll, P.

This document presents a bibliography of references on a comparatively new discipline called *domain analysis*. This discipline defines a process to identify and represent the relevant information in a *domain* (a set of systems which share common capabilities). The information is derived from:

1. The study of existing systems and their development histories
2. Knowledge captured from domain experts
3. Underlying theory
4. Emerging technology

The bibliography's purpose is to provide an historical perspective on the field as well as a necessary background for further work in the discipline.

http://www.sei.cmu.edu/publications/documents/90.reports/90.sr.003.html

Curriculum Modules and Support Materials

SEI-CM-26 ADA235700 *Understanding Program Dependencies*

Wilde, N. (University of West Florida)

August 1990

A key to program understanding is unravelling the interrelationships of program components. This module discusses the different methods and tools that aid a programmer in answering the questions "How does this system fit together?" and "If I change this component, what other components might be affected?"

http://www.sei.cmu.edu/publications/documents/cms/cm.026.html

SEI-CM-25 ADA223760 *Language and System Support for Concurrent Programming*

Feldman, M. (The George Washington University)

April 1990

This curriculum module is concerned with support for concurrent programming provided to the application programmer by operating systems and programming languages. This includes system calls and language constructs for process creation, termination, synchronization, and communication, as well as nondeterministic language constructs such as the selective wait and timed call. Several readily available languages are discussed and compared; concurrent programming using system services of the UNIX operating system is introduced for the sake of comparison and contrast.

http://www.sei.cmu.edu/publications/documents/cms/cm.025.html

SEI-SM-25 ADA223739 *Support Materials for Language and System Support for Concurrent Programming*

Ford, G. (ed.)

April 1990

This support materials package includes materials helpful in teaching concurrent programming.

http://www.sei.cmu.edu/publications/documents/cms/sm.025.html

SEI-CM-24, ADA223897

Concepts of Concurrent Programming

Bustard, D. (University of Ulster)

April 1990

A concurrent program is one defining actions that may be performed simultaneously. This module discusses the nature of such programs and provides an overview of the means by which they may be constructed and executed. Emphasis is given to the terminology used in this field and the underlying concepts involved.

http://www.sei.cmu.edu/publications/documents/cms/sm.024.html

SEI-CM-23 ADA223872

Technical Writing for Software Engineers

Levine, L.; Pesante, L.; Dunkle, S.

May 1990

This module, which is directed specifically to software engineers, discusses the writing process in the context of software engineering. Its focus is on the basic problem-solving activities that underlie effective writing, many of which are similar to those underlying software development. The module draws on related work in a number of disciplines, including rhetorical theory, discourse analysis, linguistics, and document design. It suggests techniques for becoming an effective writer and offers criteria for evaluating writing.

http://www.sei.cmu.edu/publications/documents/cms/cm.023.html

SEI-CM-19-1.2
ADA235642

Software Requirements

Brackett, J. (Boston University)

January 1990

This curriculum module is concerned with the definition of software requirements—the software engineering process of determining what is to be produced–and the products generated in that definition. The process involves all of the following: (1) requirements identification (2) requirements analysis (3) requirements representation (4) requirements communication (5) development of acceptance criteria and procedures. The outcome of requirements definition is a precursor of software design.

This module supersedes SEI-CM-1.

http://www.sei.cmu.edu/publications/documents/cms/cm.019.html

SEI-CM-11-2.1
ADA235643

Software Specification: A Framework

Rombach, H.D. (University of Maryland)

January 1990

This curriculum module presents a framework for understanding software product and process specifications. An unusual approach has been chosen in order to be able to address all aspects related to specification without confusing the many existing uses of the term. In this module, the term specification refers to any plan (or standard) according to which products of some type are constructed or processes of some type are performed, not to the products or processes themselves. In this sense, a specification is itself a product that describes how products of some type should look or how processes of some type should be performed. The framework includes (1) a reference software life-cycle model and terminology, (2) a characterizing scheme for software product and process specifications, (3) guidelines for using the characterization scheme to identify clearly certain life-cycle phases, and (4) guidelines for using the characterization scheme to select and evaluate specification techniques.

http://www.sei.cmu.edu/publications/documents/cms/cm.011.html

Educational Materials

CMU/SEI-90-EM-003
ADA228026

Reading Computer Programs: Instructor's Guide and Exercises

Deimel, L.; Naveda, J. Fernando (University of Scranton)

The ability to read and understand a computer program is a critical skill for the software developer, yet this skill is seldom developed in any systematic way in the education or training of software professionals. These materials discuss the importance of program reading, and review what is known about reading strategies and other factors affecting comprehension. These materials also include reading exercises for a modest ADA program and discuss how educators can structure additional exercises to enhance program reading skills.

http://www.sei.cmu.edu/publications/documents/ems/90.em.003.html

1989 Reports

Technical Reports

CMU/SEI-89-TR-040
ADA228034

Dark Porting and Extension Guide Kernel Version 3.0

Bamberger, J.; Coddington, T.; Firth, R.; Klein, D.; Stinchcomb, D.; Van Scoy, R.

This document describes the modifications made to the Distributed ADA Real-Time Kernel (DARK) software when porting it form its original execution environment, the 68020-based testbed built at the Software Engineering Institute, to a VAX/VMS system. This document also contains information about logical extensions to the Kernel, and the impacts thereof, should the Kernel be used in operational systems.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.040.html

CMU/SEI-89-TR-038
ADA223762

Inertial Navigation System Simulator Program: Top-Level Design

Fowler, K.

Hard-real time systems have consistently proven to be some of the most difficult for successful software implementation. Attributes often associated with the intractable nature of real-time are concurrency, severe timing constraints, the complexity of real-world devices, and limited resources. In this experiment, an actual embedded hard real-time application (Inertial Navigation Set, AN/WSN-5) is simulated and ported to a variety of target processors. The effort is specifically directed at investigating the capability of ADA for providing program development solutions in the hard real-time regime. Special emphasis is focused on applying the built-in concurrency capabilities of ADA. The effort contends with typical cross-targeting issues such as board-level execution and memory configuration, device communications, and runtime debugging of the application. This report presents the top-level design of the application and addresses the solution in terms of a concurrency abstraction. Beginning with a classical data flow analysis of the requirements, ADA tasks are derived from analyzable categories, specifically periodics, aperiodics, and servers. This classification scheme is predicated on work actively being conducted on a scheduling technique that quantifies the effect of task preemption and blocking, behavior fundamental to the concept of parallelism in ADA. In a corollary report [Borger, M. 89], a schedulability analysis of the INS is described within the framework of the task set developed in this top-level design.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.038.html

CMU/SEI-89-TR-036
ADA219187

Comparative Evaluations of Four Specification Methods for Real-Time Systems

Wood, D.; Wood, W.

A number of methods have been proposed in the last decade for the specification of system and software requirements for time-critical systems. The emerging CASE technology is based heavily on a subset of these methods; yet little *objective* attention has been paid to the methods themselves. This report describes our objective evaluation of four methods (identified as *ESML, Harel, Hatley-Pirbhai*, and *Ward-Mellor*), from identification through detailed assessment. We have avoided the use of small sample problems as the sole basis of our evaluation. We depart from this approach by involving software developers from various application domains, including extended interviews of those who have applied the methods to large-scale projects. The resulting recommendations and conclusions focus on method selection criteria, and on the large-grained impact of using these methods on a given project.

The primary audience of this report is the software development practitioner involved in the method selection or adoption process. The paper attempts to provide proper context to assist the practitioner in making appropriate method adoption decisions. Secondarily, the results of the paper also should be of to tool vendors, method developers, and program managers.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.036.html

CMU/SEI-89-TR-035
ADA219294

Inertial Navigation System Simulator: Behavioral Specification

Landherr, S.; Klein, M

The Real-Time Embedded Systems Testbed (REST) Project at the Software Engineering Institute is specifying and developing a representative real-time application. This document augments an original set of specifications written by a Navy affiliate. The purpose of this behavioral specification is to clarify and augment the original.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.035.html

CMU/SEI-89-TR-034
ADA219293

Durra: A Task-Level Description Language Reference Manual

Barbacci, M.; Wing, J.

Durra is a language designed to support the development of large-grained parallel programming applications. These applications are often computation-intensive, or have real-time requirements that require efficient concurrent execution of multiple tasks, devoted to specific pieces of the application. During execution time the application tasks run on possibly separate processors, and communicate with each other by sending messages of different types across various communication links. The application developer is responsible for prescribing a way to manage all of these resources. We call this prescription a *task-level application description*. It describes the tasks to be executed, the possible assignments of processes to processors, the data paths between the processors, and the intermediate queues required to store the data as they move from source to destination processes. Durra is a *task-level description language,* a notation in which to write these application descriptions.

This document is a revised version of the original reference manual. It describes the syntax and semantics of the language and incorporates all the language changes introduced as a result of our experiences writing task and application descriptions in Durra. A companion document, *Durra: A Task-Level Description Language User's Manual* describes how to use the compiler, runtime environment, and support tools.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.034.html

CMU/SEI-89-TR-033
ADA223761

Durra: A Task-Level Description Language User's Manual

Barbacci, M.; Doubleday, D.; Weinstock, C.

Durra is a language designed to support the development of large-grained parallel programming applications. This is the manual for users of the Durra compiler, runtime system, and support tools. Additional documents that describe the syntax and semantics of the language and the runtime environment are cited in the bibliography section.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.033.html

CMU/SEI-89-TR-032
ADA219290

Durra Application Debugger/Monitor, The

Doubleday, D.

Durra is a language designed to support the construction of distributed applications using concurrent, coarse-grained tasks running on networks of heterogeneous processors. An application written in Durra describes the tasks to be instantiated and executed as concurrent processes, the types of data to be exchanged by the processes, and the intermediate queues required to store the data as they move from producer to consumer processes.

This report describes the Durra application debugger/monitor, a program that works in conjunction with the Durra runtime software to help the developer locate errors and/or performance bottlenecks in a Durra application.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.032.html

CMU/SEI-89-TR-030
ADA219064

What a Software Engineer Needs to Know: I. Program Vocabulary

Shaw, M.; Giuse, D. (School of Computer Science, Carnegie Mellon University);
Reddy, R. (School of Computer Science, Carnegie Mellon University)

Software development, like any complex task, requires a wide variety of knowledge and skills. We examine one particular kind of knowledge, the *programming language vocabulary* of the programmer, by gathering statistics on large bodies of code in three languages. This data shows that most of the identifiers in programs are either uses of built-in or standard library definitions or highly idiomatic uses of local variables. We interpret this result in light of general results on expertise and language acquisition. We conclude that tools to support the vocabulary component of software development are wanting, and this part of an engineer's education is at best haphazard, and we recommend ways to improve the situation.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.030.html

CMU/SEI-89-TR-028
ADA219188

Understanding the Adoption of ADA: A Field Study Report

Smith, G.; Hefley, W.

In 1983, the U.S. Department of Defense (DoD) established for the development of all new DoD mission-critical computer applications. A multi-industry field study was conducted with seven business units from DoD contractors that have made decisions about the adoption and use of ADA. This report examines the extent to which the ADA adoption behavior of these contractors is influenced by their expectations of the technological opportunity provided by ADA, market demand for ADA, and appropriability conditions in the product market of firms. Findings indicate contractor decisions about adopting ADA are influenced both by the technical merits of the language and the economic impact on the firm.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.028.html

CMU/SEI-89-TR-026
ADA219066

CASE Planning and the Software Process

Humphrey, W.

Automating a software process both magnifies its strengths and accentuates its weaknesses. Automation can make an effective process more effective, but it can make a chaotic process even worse–and at considerable expense. Anyone who buys expensive tools to solve an ill-defined problem is likely to be disappointed. Unless procuring such tools is part of a thoughtful software process improvement plan, the purchase could be an expensive mistake.

This report discusses software process maturity and its relationship to planning and installing computer-aided software engineering (CASE) systems. While process is not a magic answer (there isn't one), the key issues are discussed from a process perspective, and guidelines are given for avoiding the most common pitfalls. Since CASE systems can involve significant investment, an economic justification may be necessary. The relevant financial considerations are therefore discussed, and some basic steps for producing such justifications are outlined. Finally, some key considerations for introducing and using CASE systems are discussed.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.026.html

CMU/SEI-89-TR-025
ADA226696

Classifying Software Design Methods

Wood, W.; Long, J.

A few mature and popular methods are currently being used to specify and design real-time embedded systems software, and these methods are the basis for a large number of tools automating the process. Unfortunately, some of the tools support only parts of a method, while others support a mixture of different methods. Because of the large number of tools involved, companies selecting tools for their particular needs are faced with a significant problem. As a result, the choice of tools often depends on the best salesperson rather than on the most appropriate method, leading to disappointment on the part of end users of the tools. The Software Engineering Institute has had a project underway for some time that provides a basis for selecting methods and tools. This paper describes some of the results of this effort with respect to classifying design methods for ADA-based software.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.025.html

CMU/SEI-89-TR-024
ADA219019

Temporal Logic Case Study

Wood, W.

This report is a case study applying temporal logic to specify the operation of a bank of identical elevators servicing a number of floors in a building. The goal of the study was to understand the application of temporal logic in a problem domain that is appropriate for the method, and to determine some of the strengths and weaknesses of temporal logic in this domain. The case study uses a finite state machine language to build a model of the system specification, and verifies that the temporal logic specifications are consistent using this model. The specification aspires to be complete, consistent, and unambiguous.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.024.html

CMU/SEI-89-TR-023
ADA219326

Hartstone: Synthetic Benchmark Requirements for Hard Real-Time Applications

Weiderman, N.

The purpose of this paper is to define the operational concept for a series of benchmark requirements to be used to test the ability of a system to handle hard real-time applications. Implementations of such benchmarks would be useful in evaluating scheduling algorithms, protocols, and design paradigms, as well as processors, languages, compilers, and operating systems. Several ADA programs are under development to test standard versions of the benchmark requirements and will be released into the public domain.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.023.html

CMU/SEI-89-TR-022
ADA219020

Real-Time Software Engineering in ADA: Observations and Guidelines

Borger, M.; Klein, M.

Two important aspects of developing a real-time system are controlling devices and managing concurrency. In this report, we present several techniques for controlling devices with ADA and several ADA tasking paradigms for managing concurrency. The material presented in this report is taken from our experiences in developing a real-time embedded system in ADA, and we use examples from this system to illustrate the various methods we present. We begin by describing our experiences using ADA to control devices. Specifically, we identify issues related to accessing device registers and handling interrupts, and present techniques for dealing with such issues. We then recount our experiences using ADA to manage concurrency. Specifically, we present coding paradigms for implementing periodicity and constructing synchronization mechanisms. We illustrate analytical methods for determining the schedulability of a task set. We then discuss the effect of aperiodic processing requirements on the schedulability of a task set.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.022.html

CMU/SEI-89-TR-021
ADA219018

1989 SEI Report on Graduate Software Engineering Education

Ardis, M.; Ford, G.

This annual report on graduate software engineering education describes recent SEI educational activities, including the 1988 SEI Curriculum Design Workshop. A model curriculum for a professional Master of Software Engineering degree is presented, including detailed descriptions of six core courses. Fifteen university graduate programs in software engineering are surveyed.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.021.html

CMU/SEI-89-TR-020
ADA192292

Version Description and Installation Guide

Bamberger, J.; Coddington, T.; Firth, R.; Klein, D.; Stinchcomb, D.; Van Scoy, R.

This document characterizes a specific version of the Distributed ADA Real-Time Kernel (DARK) software artifact and supplies documentation for its installation and use. This document is geared toward: the engineer responsible for installing the Kernel, engineers responsible for porting and maintaining the Kernel, and engineers using the Kernel and needing an awareness of changes from the previous release.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.020.html

CMU/SEI-89-TR-019
ADA219295

Kernel Architecture Manual

Bamberger, J.

This document contains the detailed design description of the Kernel. he overall system architecture and the rationale for it are presented as relevant to both the application (i.e., the external view of the Kernel) and the Kernel maintainer (i.e., the internal view of the Kernel). This document presents the algorithms and data structures needed to implement the functionality defined in the Kernel Facilities Definition. This document also contains an in-depth description of the communication protocol used by the Kernel, both the network software and hardware that compose the DARK testbed at the SEI, and a detailed enumeration of all compiler dependencies exploited by Kernel software. his document is geared toward engineers responsible for porting and maintaining the Kernel and engineers requiring detailed information about the internals of the Kernel.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.019.html

CMU/SEI-89-TR-018
ADA211514

Real-Time Locking Protocol, A

Sha, L.; Ragunathan, R.; Sang, S.; Chun-Hyon, C.

When a database system is used in a real-time application, the concurrency control protocol must satisfy not only the consistency of shared data but also the timing constraints of the application. In this paper, we examine a priority-driven two-phase lock protocol called the read- or write-priority ceiling protocol. We show that this protocol is free of deadlock, and in addition a high-priority transaction can be blocked by lower priority transactions for at most the duration of a single embedded transaction. We then evaluate system performance experimentally.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.018.html

CMU/SEI-89-TR-017
ADA211573

Adoption of Software Engineering Innovations in Organizations

Bayer, J.; Melone, N.

Designing effective strategies to facilitate the adoption of new software engineering technologies is a complex endeavor. This document describes the experiences of organizations in the defense industry that have considered and in many cases adopted any one of five software engineering technologies: structured programming, program design languages, software cost models, complexity metrics, and ADA. In all, 296 respondents participated in the entire study. These respondents represented approximately 120 business units within approximately 75 defense contractor organizations. Data were collected using a structured survey instrument administered over the telephone.

This report examines the motivations behind technology acquisition and adoption decisions, the use of various technology transfer mechanisms during the stages of the adoption process, and the relationship between technology transfer mechanisms and the timing, pass through, and smoothness of adoption process stages. Adoption is assumed to be a multi-stage process that may proceed in a linear or non-linear fashion. Also explored is the relationship between managerial level of the advocate (i.e., top management, middle management, technical management, and broad-based support) and the speed and smoothness of technology acquisition and adoption.

Analysis of data supports the notion that organizations and change agents (e.g., the Department of Defense) should carefully tailor transition mechanisms and the choice of technology advocate to the specific stage of the adoption process, rather than adopt a single strategy for the entire process. Moreover, a single adoption strategy is not applicable to all technologies. These strategies must also be tailored depending on the subtleties of the particular technology.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.017.html

CMU/SEI-89-TR-016
ADA228027

Guidelines for the Use of the SAME

Graham, M.

These guidelines describe the Structured Query Language (SQL) ADA Module Extensions, or SAME, a method for the construction of ADA applications that access database management systems whose data manipulation language is SQL. As its name implies, the SAME extends the module language defined in the ANSI SQL standard to fit the needs of ADA. The defining characteristic of the use of the module language is that the SQL statements appear together, physically separated from the ADA application, in an object called the module. The ADA application accesses the module through procedure calls.

The primary audience for this document consists of application developers and technicians creating ADA applications for SQL database management systems. The document contains a complete description of the SAME, including its motivation.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.016.html

CMU/SEI-89-TR-015
ADA209607

Implementing Priority Inheritance Algorithms in an ADA Runtime System

Borger, M.; Ragunathan, R. (Carnegie Mellon University)

This paper presents a high-level design–in the form of necessary data structures, mechanisms, and algorithms–for implementing the basic priority inheritance and priority ceiling protocols in an ADA runtime system. Both of these protocols solve the unbounded priority inversion problem, where a high-priority task can be forced to wait for a lower priority task for an arbitrary duration of time. The protocols and their implementation also address the issues of non-deterministic selection of open alternatives and FIFO entry call queues. These protocols allow the timing analysis of a given set of ADA tasks in order to guarantee their deadlines in real-time systems. Importantly, it is possible to implement the protocols within the current semantics of the ADA language given the interpretations of ADA rules described by Goodenough and Sha in the Software Engineering Institute Technical Report 33 (1988). Strategies and possible alternatives are discussed for implementing these protocols in an ADA runtime system targeted to a uniprocessor execution environment.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.015.html

CMU/SEI-89-TR-014
ADA211397

Real-Time Scheduling Theory and ADA

Sha, L.; Goodenough, J.

The ADA tasking model was intended to support the management of concurrency in a priority-driven scheduling environment. In this paper, we review some important results of a priority-based scheduling theory, illustrate its applications with examples, discuss its implications for the ADA tasking model, and suggest workarounds that permit us to implement analytical scheduling algorithms within the existing framework of ADA. This paper is a revision of CMU/SEI-88-TR-033. The most important revisions affect our discussion of aperiodic tasks and our analysis of how to support the priority ceiling protocol. A shortened version is also being presented at the 1989 ADA-Europe Conference.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.014.html

CMU/SEI-89-TR-013
ADA207717

ADA Adoption Handbook: Compiler Evaluation and Selection Version 1.0

Weiderman, N.

The evaluation and selection of an ADA compilation system for a project is a complex and costly process. Failure to thoroughly evaluate an ADA compilation system for a particular user application will increase project risk and may result in cost and schedule overruns. The purpose of this handbook is to convince the reader of the difficulty and importance of evaluating an ADA compilation system (even when there is no freedom of choice). The handbook describes the dimensions along which a compilation system should be evaluated, enumerates some of the criteria that should be considered along each dimension, and provides guidance with respect to a strategy for evaluation. The handbook does *not* provide a cookbook for evaluation and selection. Nor does it provide information on specific compilation systems or compare different compilation systems. Rather it serves as a reference document to inform users of the options available when evaluating and selecting an ADA compilation system.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.013.html

CMU/SEI-89-TR-012
ADA219189

Model Solution for C3I Message Translation and Validation, A

Plinta, C.; Lee, K.; Rissman, M.

This document describes an artifact, the Message Translation and Validation (MTV) model solution. The MTV model solution is a general solution, written in ADA, that can be used in a system required to convert between different message representations. These message representations can be character-based, bit-based, and internal (i.e., ADA values). This document provides designers with enough information to determine whether this solution is applicable to their particular problem. It gives detailed designers the information needed to specify solutions to their particular problem using the MTV model solution. Finally, it describes the MTV model solution in enough detail to enable a maintainer or adapter to understand the solution.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.012.html

CMU/SEI-89-TR-011
ADA211344

Scheduling Sporadic and Aperiodic Events in a Hard Real-Time System

Sprunt, B.; Sha, L.

A real-time system consists of both aperiodic and periodic tasks. Periodic tasks have regular arrival times and hard deadlines. Aperiodic tasks have irregular arrival times and either soft or hard deadlines. In this paper, we present a new algorithm, the Sporadic Server algorithm, that greatly improves response times for soft-deadline aperiodic tasks and can guarantee hard deadlines for both periodic and aperiodic tasks. The operation of the Sporadic Server algorithm, its performance, and schedulability analysis are discussed and compared with previous, published aperiodic service algorithms.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.011.html

CMU/SEI-89-TR-009
ADA206575

Command, Control, Communications, and Intelligence Node: A Durra Application Example

Barbacci, M.; Doubleday, D.; Weinstock, C.

Durra is a language designed to support the construction of distributed applications using concurrent, coarse-grain tasks running on networks of heterogeneous processors. An application written in Durra describes the tasks to be instantiated and executed as concurrent processes, the types of data to be exchanged by the processes, and the intermediate queues required to store the data as they move from producer to consumer processes. This report describes an experiment in implementing a command, control, communications and intelligence (C^3I) node using reusable components. The experiment involves writing task descriptions and type declarations for a subset of the TRW testbed, a collection of C^3I software modules developed by TRW Defense Systems Group. The experiment illustrates the development of a typical Durra application. This is a three-step process: first, a collection of tasks (programs) is designed and implemented (these are the testbed programs); second, a collection of task descriptions

corresponding to the task implementations is written in Durra, compiled, and stored in a library; and finally, an application description is written in Durra and compiled, resulting in a set of resource allocation and scheduling commands to be interpreted at runtime.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.009.html

CMU/SEI-89-TR-008
ADA207415

Performance and Reliability Enhancement of the Durra Runtime Environment

Weinstock, C.

Durra is a language designed to support PMS-level programming. PMS stands for Processor Memory Switch, the name of the highest level in the hierarchy of digital systems. An application or PMS-level program is written in Durra as a set of *task descriptions* and *type declarations* that prescribes a way to manage the resources of a heterogeneous machine network. The application describes the tasks to be instantiated and executed as concurrent processes, the types of data to be exchanged by the processes, and the intermediate queues required to store the data as they move from producer to consumer processes.

A runtime environment for Durra has been operational for some time. There are two major problems with this initial implementation: it makes no significant attempt to tune the performance of the system, and reliability has not been designed into the system. This report describes a new design for the Durra runtime environment that addresses these two issues. The new runtime environment consists of two major components: a *local executive* which runs on every processor and is responsible for process and queue management, and a *global executive* which runs replicated on several processors and is responsible for configuration management and reliability services.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.008.html

CMU/SEI-89-TR-007
ADA219065

Conducting SEI-Assisted Software Process Assessments

Olson, T.; Humphrey, W.; Kitson, D.

This report describes software process assessment as it is performed in organizations with the assistance of the Software Engineering Institute. A software process assessment is an appraisal or review of an organization's software process (e.g., software development process). The main objectives of such an assessment are to understand the state of practice in an organization, to identify key areas for improvement, and to initiate the actions that facilitate those improvements. This report is specifically addressed to the organizations and assessment team members that may be involved in an SEI-assisted software process assessment.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.007.html

CMU/SEI-89-TR-006
ADA206779

Proceedings of the Workshop on Executive Software Issues August 2-3 and November 18, 1988

Martin, D.; Carey, S.; Coticchia, M.; Fowler, P.; Maher, J.

These proceedings document the results of two sessions of the Workshop on Executive Software Issues held at the Software Engineering Institute on August 2-3 and November 18, 1988. The purpose of the workshop was to define the issues that should be brought to the attention of executives in the defense industry, government, and academia and to propose resolutions to those issues. The issues discussed were divided into three broad categories: national strategy, acquisition, and building large complex systems. Described in the proceedings are the major issues and recommended actions.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.006.html

CMU/SEI-89-TR-005
ADA219190

Object-Oriented Solution Example: A Flight Simulator Electrical System, An

Lee, K.; Rissman, M.

This report describes an implementation of a subset of an aircraft flight simulator electrical system. It is a result of work on the ADA Simulator Validation Program (ASVP) carried out by members of the technical staff (MTS) at the SEI. The MTS developed a paradigm for describing and implementing flight simulator systems in general. The paradigm is a model for implementing systems of objects. The objects are described in a form of design specification called an object diagram. This report has been prepared to demonstrate a full implementation of a system: package specifications and bodies. The intent is to provide an example; the code is functional, but there is no assurance of robustness.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.005.html

CMU/SEI-89-TR-004
ADA206574

Human-Machine Interaction Considerations for Interactive Software

Bass, L.; Coutaz, J.

This document introduces current concepts and techniques relevant to the design and implementation of user interfaces. A user interface refers to those aspects of a system that the user refers to, perceives, knows, and understands. A user interface is implemented by code that mediates between a user and a system. This document covers both aspects.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.004.html

CMU/SEI-89-TR-003
ADA227426

Role of Assessment in Software Process Improvement, The

Kitson, D.; Humphrey, W.

This report discusses the role of assessment in improving an organization's software capabilities; specifically, the ability of the organization's projects to consistently meet cost, schedule, and quality objectives. Software process assessments are described from both a conceptual and pragmatic point of view. Underlying concepts of software process, software process management, and software process maturity are discussed. Collectively, these constitute a framework for software process assessment and improvement.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.003.html

CMU/SEI-89-TR-002
ADA211636

Software Process Modeling: Principles of Entity Process Models

Humphrey, W.; Kellner, M.

A defined software process is needed to provide organizations with a consistent framework for performing their work and improving the way they do it. An overall framework for modeling simplifies the task of producing process models, permits them to be tailored to individual needs, and facilitates process evolution. This paper outlines the principles of entity process models and suggests ways in which they can help to address some of the problems with more conventional approaches to modeling software processes.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.002.html

CMU/SEI-89-TR-001
ADA206573

State of Software Engineering Practice: A Preliminary Report, The

Humphrey, W.; Kitson, D.; Kasse, T.

This is the first in a series of SEI reports to provide periodic updates on the state of software engineering practice in the DoD software community. The SEI has developed, and is refining, a process framework and assessment methodology for characterizing the processes used by software organizations to develop and evolve software

products. This report provides a brief overview of the process framework and assessment approach, describes assessment results obtained to date, and discusses implications of the current state of the practice for both customers and suppliers of DoD software.

http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.001.html

Special Reports

CMU/SEI-89-SR-018
ADA275239

Recommendations from the AIA/SEI Workshop on Research Advances Required for Real-Time Software Systems in the 1990s

Sweet, W.; Gagliardi, M.; Klein, M.; Little, R.; Van Scoy, R.; Veltre, R.; Weinstock, C.

The Aerospace Industries Association (AIA) and the Software Engineering Institute sponsored a workshop to facilitate clear communication between the implementors of future software-critical large systems and the professionals who sponsor or perform software-related research. The workshop was held at the SEI in Pittsburgh, Pennsylvania, on September 13-14, 1989. At the workshop, a representative group of designers of major upcoming software systems discussed among themselves and with representatives of the research community the areas of software system technology that require research advances to successfully implement these systems. The results of the discussions are summarized in this report.

CMU/SEI-89-SR-014
ADA250349

Conformance Criteria for the SAME Approach to Binding ADA Programs to SQL

Moore, J.

The structured query language (SQL) ADA Module Extensions (SAME) form a method for the design and construction of ADA database applications. The method is explained in a companion document: *Guidelines for the Use of the SAME*. In order to enable the method to be referenced in requests for proposals (RFPs) and development contracts, there must be some method to determine if a given software design and implementation conform to the SAME guidelines. Such conformance criteria are contained in this report.

http://www.sei.cmu.edu/publications/documents/89.reports/89.sr.014.html

CMU/SEI-89-SR-005
ADA253172

SAME Standard Package Installation Guide

Graham, M.

This document outlines the procedures for installing the SQL ADA Module Extensions (SAME) standard packages. It assumes that the source of these packages has been off-loaded from the distribution medium to a machine with the desired ADA compiler.

http://www.sei.cmu.edu/publications/documents/89.reports/89.sr.005.html

Curriculum Modules and Support Materials

SEI-CM-22-1.0
ADA235701

Software Design Methods for Real-Time Systems

Gomaa, H. (George Mason University)

December 1989

This module describes the concepts and methods used in the software design of real-time systems. It outlines the characteristics of real-time systems, describes the role of software design in real-time system development, surveys and compares some software design methods for real-time systems, and outlines techniques for the verification and validation of real-time designs. For each design method treated, its emphasis, concepts on which it is based, steps used in its application, and an assessment of the method are provided.

http://www.sei.cmu.edu/publications/documents/cms/cm.022.html

SEI-CM-21-1.0
ADA237048

Software Project Management

Tomayko, J. (The Wichita State University); Hallman, H.K.

July 1989

Software project management encompasses the knowledge, techniques, and tools necessary to manage the development of software products. This curriculum module discusses material that managers need to create a plan for software development, using effective estimation of size and effort, and to execute that plan with attention to productivity and quality. Within this context, topics such as risk management, alternative life-cycle models, development team organization, and management of technical people are also discussed.

http://www.sei.cmu.edu/publications/documents/cms/cm.021.html

SEI-CM-17-1.1
ADA235699

User Interface Development

Perlman, G. (Massachusetts Institute of Technology)

November 1989

This module covers the issues, information sources, and methods used in the design, implementation, and evaluation of user interfaces, the parts of software systems designed to interact with people. User interface design draws on the experiences of designers, current trends in input/output technology, cognitive psychology, human factors (ergonomics) research, guidelines and standards, and on the feedback from evaluating working systems. User interface implementation applies modern software development techniques to building user interfaces. User interface evaluation can be based on empirical evaluation of working systems or on the predictive evaluation of system design specifications.

http://www.sei.cmu.edu/publications/documents/cms/cm.016.html

SEI-CM-16-1.1 ADA235-996

Software Development Using VDM

Storbank Pedersen, J. (Computer Resources International A/S)

December 1989

This module introduces the Vienna Development Method (VDM) approach to software development. The method is oriented toward a formal model view of the software to be developed. The emphasis of the module is on formal specification and systematic development of programs using VDM. A major part of the module deals with the particular specification language (and abstraction mechanisms) used in VDM.

http://www.sei.cmu.edu/publications/documents/cms/cm.016.html

SEI-CM-14-2.1
ADA236125

Intellectual Property Protection For Software

Samuelson, P. (University of Pittsburgh School of Law);
Deasy, K. (University of Pittsburgh School of Law)

July 1989

This module provides an overview of the U.S. intellectual property laws that form the framework within which legal rights in software are created, allocated, and enforced. The primary forms of intellectual property protection that are likely to apply to software are copyright, patent, and trade secret laws, which are discussed with particular emphasis on the controversial issues arising in their application to software. Also included is a brief introduction to government software acquisition regulations, trademark, trade dress, and related unfair competition issues that may affect software engineering decisions, and to the Semiconductor Chip Protection Act.

http://www.sei.cmu.edu/publications/documents/cms/cm.014.html

SEI-CM-9-1.2
ADA236119

Unit Testing and Analysis

Morell, L.J. (College of William and Mary)

April 1989

This module examines the techniques, assessment, and management of unit testing and analysis. Testing and analysis strategies are categorized according to whether their coverage goal is functional, structural, error-oriented, or a combination of these. Mastery of the material in this module allows the software engineer to define, conduct, and evaluate unit tests and analyses and to assess new techniques proposed in the literature.

http://www.sei.cmu.edu/publications/documents/cms/cm.009.html

SEI-SM-3-1.0
ADA236122

Support Materials for The Software Technical Review Process

Cross, J. (ed.) (Indiana University of Pennsylvania)

April 1989

This support materials package includes materials helpful in teaching a course on the software technical review process.

http://www.sei.cmu.edu/publications/documents/cms/sm.003.html

SEI-CM-2-2 ADA236118

Introduction to Software Design

Budgen, D. (University of Stirling)

January 1989

This curriculum module provides an introduction to the principles and concepts relevant to the design of large programs and systems. It examines the role and context of the design activity as a form of problem-solving process, describes how this is supported by current design methods, and considers the strategies, strengths, limitations, and main domains of application of these methods.

http://www.sei.cmu.edu/publications/documents/cms/cm.002.html

Educational Materials

CMU/SEI-89-EM-002
A235777

APSE Interactive Monitor: A Software Artifact for Software Engineering Education

Engle, C.; Ford, G.; Tomayko, J.

In 1987 the SEI began a search for a well-documented ADA system, developed under government contract, that could be used in software engineering education. The APSE Interactive Monitor (AIM) was determined to be appropriate for this purpose. This system acts as an interface between a user of an ADA programming support environment (APSE) and the programs that the user executes in the APSE. It provides facilities to support the concurrent execution of multiple interactive programs, each of which has access to a virtual terminal. Educational uses of the system are described, including use as a case study and as the basis for exercises. Software engineering topics that can be taught with the system include software maintenance, configuration management, software documentation, cost estimation, and object-oriented design.

http://www.sei.cmu.edu/publications/documents/ems/89.em.002.html

CMU/SEI-89-EM-001
ADA235779

Software Maintenance Exercises for a Software Engineering Project Course

Engle, C.; Ford, G.; Korson, T.

This report provides an operational software system of 10,000 lines of ADA and several exercises based on that system. Concepts such as configuration management, regression testing, code reviews, and stepwise abstraction can be taught with these exercises.

http://www.sei.cmu.edu/publications/documents/ems/89.em.001.html

1988 Reports

Technical Reports

CMU/SEI-88-TR-035

Experiment Planning for Software Development: Redevelopment Experiment

J.M. Perry, K.C. Kang, S. Cohen, R. Holibaugh, A.S. Peterson

The Application of Reusable Software Components Project (ARSC) formulated an experiment design, data collection plan, and procedures in preparation for a reuse experiment. The reuse experiment is currently in progress and the experiment planning and the results to date are presented. While the design, plan, and procedures were developed to support the investigation of software reuse, they, as well as the process by which they were formulated, are applicable to any software development effort. They can be adapted to other technology investigations or to project-specific goals for improvement.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.035.html

CMU/SEI-88-TR-034
ADA20754

Mode Change Protocols for Priority-Driven Preemptive Scheduling

Sha, L.; Goodenough, J.

In many real-time applications, the set of tasks in the system as well as the characteristics of the tasks change during system execution. Specifically, the system moves from one mode of execution to another as its mission progresses. A mode change is characterized by the deletion of some tasks, addition of new tasks, or changes in the parameters of certain tasks, e.g., increasing the sampling rate to obtain a more accurate result. This paper discusses a protocol for systematically accomplishing mode change in the context of a priority-driven preemptive scheduling environment.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.034.html

CMU/SEI-88-TR-033
ADA205048

Real-Time Scheduling Theory and ADA

Sha, L.; Goodenough, J.

The ADA tasking model was intended to facilitate the management of concurrency in a priority-driven scheduling environment. In this paper, we will review some important results of a priority-based scheduling theory, illustrate its applications with examples, discuss its implications for the ADA tasking model, and suggest workarounds that permit us to implement analytical scheduling algorithms within the existing framework of ADA.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.033.html

CMU/SEI-88-TR-030
ADA204849

OOD Paradigm for Flight Simulators, 2nd Edition, An

Lee, K.; Rissman, M.; D'Ippolito, R.; Plinta, C.; Van Scoy, R.

This report presents a paradigm for object-oriented implementation of flight simulators. It is a result of work on the ADA Simulator Validation Program (ASV) carried out by members of the technical staff at the SEI.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.030.html

CMU/SEI-88-TR-026
ADA204757

Using the Vienna Development Method (VDM) to Formalize a Communication Protocol

Pedersen, J.; Klein, M.

The Vienna Development Method (VDM) is based upon iterative refinement of formal specifications written in the model-oriented specification language, Meta-IV. VDM is also an informal collection of experiences in formal specification within several application domains. This paper provides an example of how VDM might be used in the area of communications, a new domain for VDM.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.026.html

CMU/SEI-88-TR-025
ADA200611

Functional Performance Specification for an External Computer System Simulator

Meyers, C.; Mumm, H.

This document defines the functional and performance requirements for the external computer system (ECS) simulator that interfaces with the inertial navigation system simulator. Both the ECS simulator and the INS simulator are being developed in ADA by the Real-Time Embedded Systems Testbed Project at the Software Engineering Institute. The ECS simulator is similar to a real-world ECS, but has reduced functionality. This document provides specifications for the major functions of the ECS simulator.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.025.html

CMU/SEI-88-TR-024
ADA223958

System Specification Document: Shipboard Inertial Navigation System Simulator and External Computer

Meyers, C.; Weiderman, N.

This document specifies high-level requirements for a shipboard inertial navigation system (INS) simulator and an external computer system that will interface with the inertial navigation system.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.024.html

CMU/SEI-88-TR-023
ADA204850

Functional Performance Specification for an Inertial Navigation System

Meyers, C.; Weiderman, N.

This document defines the functional and performance requirements for the inertial navigation system simulator that interfaces with the external computer system (ECS) simulator. Both the INS simulator and the ECS simulator are being developed in ADA by the Real-Time Embedded Systems Testbed Project at the Software Engineering Institute. The INS simulator is similar to a real-world INS, but has reduced functionality. This document provides specifications for the major functions of the INS simulator.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.023.html

CMU/SEI-88-TR-022
ADA204399

Perspective on Software Reuse

Perry, J.

This report presents a perspective on software reuse in the context of “ideal” software development capabilities. Software reuse is viewed as a means of achieving—or at least approximating—the idea capabilities. A generic application and development model is proposed for unifying various types of software reuse. The model can be initially formulated as a project family architecture and produced from a domain features analysis. The approach presented in this report is intended to lead to a reuse strategy and methodology for software development.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.022.html

CMU/SEI-88-TR-021
ADA204634

Experiment Transcripts for the Evaluation of the Rational Environment

Downey, G.; Bassman, M.; Dahlke, C. (Computer Sciences Corporation)

This report supplements the report *Evaluation of the Rational Environment* (CMU/SEI-88-TR-015). It contains the instantiation of the experiments presented in the *Evaluation of ADA Environments* by Nelson Weiderman, N., et al. (CMU/SEI-87-TR-001). Overall conclusions and analysis of the Rational Environment can be found in *Evaluation of the Rational Environment*.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.021.html

CMU/SEI-88-TR-019
ADA199481

Generalized Image Library: A Durra Application Example

Barbacci, M.; Doubleday, D.

Durra is a language designed to support the construction of distributed applications using concurrent, coarse-grain tasks running on networks of heterogeneous processors. An application written in Durra describes the tasks to be instantiated and executed as concurrent processes, the types of data to be exchanged by the processes, and the intermediate queues required to store the data as they move from producer to consumer processes.

This report describes an experiment in writing task descriptions and type declarations for a subset of the Generalized Image Library, a collection of utilities developed at the Department of Computer Science at Carnegie Mellon University. The experiment illustrates the development of a typical Durra application. This is a three step process: first, a collection of tasks (programs) is designed and implemented (these are the GIL programs); second, a collection of task descriptions corresponding to the task implementations is written in Durra, compiled, and stored in a library; and finally, an application description is written in Durra and compiled, resulting in a set of resource allocation and scheduling commands to be interpreted at runtime. A few sample application descriptions were developed as part of the experiment and are also reported in this document.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.019.html

CMU/SEI-88-TR-018
ADA199480

Durra Runtime Environment, The

Barbacci, M.; Doubleday, D.; Weinstock, C.

Durra is a language designed to support PMS-level programming. PMS stands for Processor-Memory-Switch, the name of the highest level in the hierarchy of digital systems. An application or PMS-level program is written in Durra as a set of *task descriptions* and *type declarations* that prescribes a way to manage the resources of a heterogeneous machine network. The application describes the tasks to be instantiated and executed as concurrent processes, the types of data to be exchanged by the processes, and the intermediate queues required to store the data as they move from producer to consumer processes.

This report describes the Durra Runtime Environment. The environment consists of three active components: the application tasks, the Durra server, and the Durra scheduler. After compiling the type declarations, the component task descriptions, and the application description, the application can be executed by starting an instance of the server on each processor, starting an instance of the scheduler on one of the processors, and downloading the component *task implementations* (i.e., the programs) to the processors. The scheduler receives as an argument the name of the file containing the scheduler program generated by the compilation of the application description. This step initiates the execution of the application.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.018.html

CMU/SEI-88-TR-017
ADA199482

Distributed ADA Real-Time Kernel

Bamberger, J.; Coddington, T.; Firth, R.; Klein, D.; Stinchcomb, D.; Van Scoy, R.; Colket, C.

This paper addresses two distinct needs of real-time applications: distribution and hard real-time scheduling mechanisms. Specifically, this paper rejects both the notion of modifying the ADA language to achieve needed real-time solutions and the current fad of extensively modifying the ADA compiler and/or vendor-supplied runtime

system. Instead, this paper defines the functionality of a Distributed ADA Real-time kernel (hereafter called the Kernel). The goal of the Kernel is to support effectively the execution of distributed, real-time ADA applications in an embedded computer environment by returning control to the user, where it belongs.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.017.html

CMU/SEI-88-TR-016
ADA198933

Kernel Facilities Definition

Bamberger, J.; Coddington, T.; Firth, R.; Klein, D.; Stinchcomb, D.; Van Scoy, R.; Colket, C.

This document defines the conceptual design of the Kernel by specifying 1) the underlying models, assumptions, and 2) restrictions that govern the design and implementation of the Kernel; and the behavioral and performance requirements to which the Kernel is built. This document is the requirements and top level design document for the Kernel.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.016.html

CMU/SEI-88-TR-015
ADA198934

Evaluation of the Rational Environment

Feiler, P.; Dart, S.; Downey, G.

This report presents an analysis of the Rational R1000 Development System for ADA, also called the Rational Environment. The evaluation combined the use of the Software Engineering Institute methodology for evaluation of ADA environments, an analysis of functionality not covered by that methodology, and an assessment of the novel environment architecture of the Rational Environment. In addition to this report, Experiment Transcripts for the Evaluation of the Rational Environment, by Grace Downey, Mitchell Bassman, and Carl Dahlke (CMU/SEI-88-TR-021) contains support material for the experimental results. The support material is the result of performing experiments based on the SEI's environment evaluation methodology. It consists of transcripts of the experiments, the detailed answers to the evaluative questions, and the detailed performance results.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.015.html

CMU/SEI-88-TR-013
ADA223895

Phase I Testbed Description: Requirements and Selection Guidelines

Holibaugh, R.; Perry, J.

The Application of Reusable Software Components Project has constructed a reuse testbed for conducting software engineering experiments in software reusability. The hardware and system software of the testbed will provide a distributed computing environment with file-server capability for the storage of reusable components and other artifacts of the development process. The testbed will support a variety of domain-independent and domain-dependent reusable components. The testbed will also support tools that foster reuse. This document contains the requirements and selection criteria for the testbed hardware, software, reusable resources, and an environment. For each of these four testbed resources, the requirements are grouped into five areas: support of experiments, maximization of experience and reusability, applicability to problem domains, acceleration of technology transition, and advancing the state of the practice in reuse.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.013.html

CMU/SEI-88-TR-011
ADA197671

Managing Development of Very Large Systems: Implications for Integrated Environment Architectures

Feiler, P.; Smeaton, R.

Version and configuration control are mechanisms for managing source code and system builds. In the development of very large systems, built by large teams, development management is the dominant factor. In this paper we examine management support for development through integrated environments and investigate the implications for environment architectures. We do so by defining a project scenario that is to be performed with integrated project

support environments. The scenario has been carefully designed to not only determine the scope of management functionality provided by a particular environment, but also to probe implications for the architecture of environments. The implications discussed in this paper are: focus on user activities; the integration of project management and development support concepts; the ability to reinforce and avoid conflict with particular organizational models; the ability to support evolution and change of the product, environment, and organization; and the capability for adaptation and insertion into a work environment. The scenario is part of a methodology for evaluation of environments currently used at the Software Engineering Institute.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.011.html

CMU/SEI-88-TR-009
ADA197137

Software Process Modeling

Kellner, M.; Hansen, G.

An SEI objective is to provide leadership in software engineering and in the transition of new software engineering technology into practice. This paper discusses a software process modeling case study conducted at the SEI.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.009.html

CMU/SEI-88-TR-008
ADA197416

Guide to the Assessment of Software Development Methods, A

Wood, W.; Pethia, R.; Roberts Gold, L.; Firth, R.

Over the past decade, the term “software engineering methods” has been attached to a variety of procedures and techniques that attempt to provide an orderly, systematic way of developing software. Existing methods approach the task of software engineering in different ways. Deciding which methods to use to reduce development costs and improve the quality of products is a difficult task. This report outlines a five-step process and an organized set of questions that provide method assessors with a systematic way to improve their understanding of and form opinions about the ability of existing methods to meet their organization’s needs.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.008.html

CMU/SEI-88-TR-007
ADA197490

Project Management Experiment, The

Feiler, P.; Smeaton, R.

This report covers a project management (PM) experiment, one of six experiments that examine different functional areas of ADA programming environments. The PM experiment was designed as part of the Evaluation of ADA Environments Project. This report describes the environment-independent part of the experiment: the activities covering the functional area, the evaluation criteria, and an experiment scenario to be performed on different environments. The experiment as it stands has been validated through internal and external review and through application to several environments that support project management.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.007.html

CMU/SEI-88-TR-006
ADA196664

Serpent Runtime Architecture and Dialogue Model

Bass, L.; Hardy, E.; Hoyt, K.; Little, R.; Seacord, R.

This paper describes the runtime architecture and dialogue model of the Serpent User Interface Management System (UIMS). Serpent uses existing software systems to create a UIMS based on a structured production model to specify the dialogue, and uses a database approach for communication between its internal layers. The model for the dialogue in Serpent supports simultaneity of subdialogues and presents the dialogue specifier with a model that views data as mapping from the application to the presentation. The database approach for communication between the layers provides a model that application programmers understand well and find easy to use. The approach also provides the power necessary to decouple the application structures from the structures implicit in the user interface.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.006.html

CMU/SEI-88-TR-005
ADA200085

Introduction to the Serpent User Interface Management System

Bass, L.; Hardy, E.; Hoyt, K.; Little, R.; Seacord, R.

Serpent is an example of the class of systems known as a User Interface Management System. It uses the X Window System to interact with the end user, and is useful both as a portion of a production system and as a separate prototyping tool. Serpent supports the development and execution of the user interface of a system. It provides an editor with which to specify the user interface and a runtime system that communicates with the application to get the data to display. The system then uses the specification previously output from the editor to decide how to display that data. This report provides a technical overview of Serpent, its components, the module used in specifying the user interface, and the editor used in constructing the user interface.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.005.html

CMU/SEI-88-TR-004
ADA197136

Evaluation and Recommendations for Technology Insertion into Technical Order Maintenance

Hansen, G.; Over, J.

As the need for mission-critical software systems increases, Post Deployment Software Support (PDSS) activities will require increased priority in planning. PDSS is "the sum of all activities required to ensure that, during the production/deployment phase of a mission-critical computer system's life, the implemented and fielded software/ system continues to support its original missions, and subsequent mission modifications and product improvements. PDSS, therefore, includes not only software "maintenance" but also the activities required for overall system support.

The SEI recognizes the importance of PDSS activities in the life cycle of mission-critical systems. In March 1986, SEI personnel met with representatives of the Air Force Logistics Command (AFLC) at Ogden Air Logistics Center (OO-ALC), Hill Air Force Base, Utah, to determine if there were areas in PDSS that the SEI could address. The AFLC representatives described the activities performed at Air Logistics Centers and problems encountered in those activities. As a result of this meeting, the SEI authorized a feasibility study to determine how it might best interact with the PDSS community. This report, written in August 1987, describes the evaluation process and the ensuing recommendations for technology insertion into technical order maintenance.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.004.html

CMU/SEI-88-TR-003
ADA201345

ISTAR Evaluation

Graham, M.; Miller, D.

ISTAR is an integrated project support environment produced by Imperial Software Technology, Ltd. This evaluation of ISTAR is intended for software technologists considering the adoption of an integrated project support environment. Researchers and others interested in environments and evaluation methods will also benefit from this report.

http://www.sei.cmu.edu/publications/documents/88.reports/88.tr.003.html

Special Reports

CMU/SEI-88-SR-004

Priority Ceiling Protocol: A Method for Minimizing the Blocking of High-Priority ADA Tasks, The

Goodenough, J.; Sha, L.

The priority ceiling protocol is a new technique that addresses the priority inversion problem, i.e., the possibility that a high-priority task can be delayed by a low-priority task. Under the priority ceiling protocol, a high priority task can be blocked at most once by a lower priority task. This paper defines how to apply the protocol to ADA. In particular, restrictions on the use of task priorities in ADA are defined as well as restrictions on the use of ADA tasking constructs. An extensive example illustrating the behavior guaranteed by the protocol is given.

This paper was presented at the 2nd International Workshop on Real-Time ADA Issues in May 1988.

http://www.sei.cmu.edu/publications/documents/88.reports/88.sr.004.html

CMU/SEI-88-SR-003
ADA206429

Protocol in a Real-Time System, A

Goodenough, J.; Locke, D.

This paper briefly discusses some of the real-time design issues that arise when using the priority ceiling protocol for real-time systems. The paper shows a small real-time system and shows how the code in the system could be structured to satisfy the requirements of the ceiling protocol.

This paper was presented at the 2nd International Workshop on Real-Time ADA Issues in May 1988.

http://www.sei.cmu.edu/publications/documents/88.reports/88.sr.003.html

CMU/SEI-88-SR-002
ADA206391

Conference Report: Overcoming the Disincentives to Modernization in the Defense Industry

Hefley, W.

The Procurement Committee of the National Security Industrial Association held its annual winter meeting on 7-10 February 1988. This conference report documents the keynote address by Thomas J. Murrin, Distinguished Service Professor in Management and Technology, Carnegie Mellon University, along with panel sessions on the topic of “Regulatory and Statutory Direction of Acquisition” and “Financial Aspects of Disincentives to Modernization.

http://www.sei.cmu.edu/publications/documents/88.reports/88.sr.002.html

Curriculum Modules and Support Materials

SEI-CM-20-1.0
ADA235775

Formal Verification of Programs

Berztiss, A. (University of Pittsburgh), Ardis, M.A.

December 1988

This module introduces formal verification of programs. It deals primarily with proofs of sequential programs, but also with consistency proofs for data types and deduction of particular behaviors of programs from their specifications. Two approaches are considered: verification after implementation that a program is consistent with its specification, and parallel development of a program and its specification. An assessment of formal verification is provided.

http://www.sei.cmu.edu/publications/documents/cms/cm.020.html

SEI-SM-17-1.0
ADA235835

Support Materials for User Interface Development

Deimel, L. (ed.)

April 1988

This support materials package includes materials helpful in teaching a course on user interface development.

http://www.sei.cmu.edu/publications/documents/sms/cm.017.html

SEI-CM-13-1.1
ADA236117

Introduction to Software Verification and Validation

Collofello, J. (Arizona State University)

December 1988

Software verification and validation techniques are introduced and their applicability discussed. Approaches to integrating these techniques into comprehensive verification and validation plans are also addressed. This curriculum module provides an overview needed to understand in-depth curriculum modules in the verification and validation area.

http://www.sei.cmu.edu/publications/documents/cms/cm.013.html

SEI-CM-12-1.1
ADA236140

Software Metrics

Mills, E. (Seattle University)

December 1988

Effective management of any process requires quantification, measurement, and modeling. Software metrics provide a quantitative basis for the development and validation of models of the software development process. Metrics can be used to improve software productivity and quality. This module introduces the most commonly used software metrics and reviews their use in constructing models of the software development process. Although current metrics and models are certainly inadequate, a number of organizations are achieving promising results through their use. Results should improve further as we gain additional experience with various metrics and models.

http://www.sei.cmu.edu/publications/documents/cms/cm.012.html

SEI-CM-3-1.5
ADA236139

Software Technical Review Process, The

Collofello, J. (Arizona State University)

June 1988

This curriculum module consists of a comprehensive examination of the technical review process in the software development and maintenance life cycle. Formal review methodologies are analyzed in detail from the perspective of the review participants, project management and software quality assurance. Sample review agendas are also presented for common types of reviews. The objective of the module is to provide the student with the information necessary to plan and execute highly efficient and cost effective technical reviews.

http://www.sei.cmu.edu/publications/documents/cms/cm.003.html

1987 Reports

Technical Reports

CMU/SEI-87-TR-048
ADA199634

Interfacing ADA and SQL

Engle, C.; Firth, R.; Graham, M.; Wood, W.

The SEI was asked by the ADA Joint Program Office to investigate the problem of interfacing programs written in ADA with database management systems implementing the SQL database language. The authors decided to concentrate on a description of the problems involved in producing an interface that would be worthy of becoming a standard. This document is meant to assist the reader in answering the question "What constitutes a good interface between ADA and SQL?" The document should be useful both in the production of a standard and in the analysis of any proposed standard.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.048.html

CMU/SEI-87-TR-047
ADA188922

IDL: Background and Status

Stone, D.; Nestor, J.

This paper presents an overview of the Interface Description Language (IDL). We describe the language and its history. We also discuss the status of the IDL community.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.047.html

CMU/SEI-87-TR-046
ADA188924

Evolving Persistent Objects in a Distributed Environment

Nestor, J.

This paper considers a class of objects, called incrementally mutable objects, that are intermediate between mutable and immutable objects. Intuitively, the only permitted modifications to an incrementally mutable object are those that add new information to the object while preserving existing information. Changes to incrementally mutable objects do not require central synchronization. When a network becomes partitioned, the same incrementally mutable object can be safely modified in each subnetwork. A mutable object can be modeled by a set of immutable objects that represent each value of the object over time and an incrementally mutable object that relates each immutable object to its successor. Multiple successors are permitted to represent parallel changes.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.046.html

CMU/SEI-87-TR-045
ADA188923

Views for Evolution in Programming Environments

Nestor, J.

Programming environments have become a focal point for much of the work directed toward improving the practice of software engineering. Such environments must provide mechanisms for recording and organizing the complex set of persistent technical and management data associated with all parts of the life cycle of large software systems. This paper focuses on one important aspect of such persistent data: how to allow evolution when the existing information must be preserved without change to maintain history. First, the role of history in programming environments is discussed. Next, the additional demands of evolution are considered and shown to lead to a set of problems. View mechanisms are suggested as a solution to the problems. A simple example involving file system directory structure is presented to illustrate these problems. A simple view mechanism, called multidirectories, is introduced and shown to solve the illustrated problems.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.045.html

CMU/SEI-87-TR-044
ADA188927

Report on the SEI Workshop on ADA in Freshman Courses

Ford, G. (ed.)

The Undergraduate Software Engineering Education Project of the SEI Education Program sponsored a workshop on ADA in Freshman Courses in June 1987. The workshop brought together several educators to discuss how the software engineering content of beginning programming and data structures courses might be improved. This report describes the workshop and summarizes the discussions and conclusions; and it also includes the position papers prepared by the participants.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.044.html

CMU/SEI-87-TR-042
ADA199877

Issues in Software: A Blue Two Visit Feasibility Assessment

Hefley, W.

The SEI participated in a series of fact-finding meetings sponsored by the Air Force Coordinating Office for Logistics Research to gather information necessary to set the scope for and to implement one or more Blue Two Visits on software. The purpose of a Blue Two Visit (BTV) is to introduce to industry's top design engineers and program managers the day-to-day constraints Air Force maintainers face on front-line operations bases. The participants experience first-hand the effects of design on maintenance. This exposure has been significant in bridging the gap between DoD and industry in understanding, documenting, and supporting Air Force weapon system requirements to increase combat supportability. This report documents discussions that attempt to address the following questions for a software-oriented BTV: 1) Do software maintainers and users have messages for software designers and programmers? 2) What are these messages? 3) How can these messages be best communicated? 4) To whom should these messages be targeted? 5) What should the BTV be called?

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.042.html

CMU/SEI-87-TR-041
ADA200606

Classification Scheme for Software Development Methods, A

Firth, R.; Wood, W.; Pethia, R.; Roberts Gold, L.; Mosley, V.; Dolce, T.

Software development methods are used to assist with the process of designing software for real-time systems. Many such methods have come into practice over the last decade, and new methods are emerging. These new methods are more powerful than the old ones, especially with regard to real-time aspects of the software. This report describes a classification scheme for software development methods, includes descriptions of the major characteristics of such methods, and contains some words of advice on choosing and applying such methods.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.041.html

CMU/SEI-87-TR-040
ADA200610

ADA Performance Benchmarks on the Motorola MC68020: Summary and Results

Donohoe, P.

This report documents the results obtained from running the ACM SIGADA Performance Issues Working Group (PIWG) and the University of Michigan ADA performance benchmarks on a Motorola MC68020 microprocessor (MVME133 VME module Monoboard Microcomputer), using the Systems Designers ADA-Plus, the TeleSoft TeleGen2, and the Verdix VAX/VMS hosted cross-compilers. A brief description of the benchmarks and the test environment is followed by a discussion of some problems encountered and lessons learned. Wherever possible, the output of each benchmark program is also included.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.040.html

CMU/SEI-87-TR-039
ADA191095

Prototype Real-Time Monitor: ADA Code

Van Scoy, R.

This report documents the ADA code of the prototype real-time monitor (RTM).

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.039.html

CMU/SEI-87-TR-038
ADA188931

Prototype Real-Time Monitor: Design

Van Scoy, R.; Plinta, C.; D'Ippolito, R.; Lee, K.; Rissman, M.

This report describes the software design used to implement the prototype real-time monitor requirements. The design is presented at three levels: system level, object level, and package architecture level. The report concludes with a discussion of the key implementation obstacles that had to be overcome to develop a working prototype: determining system addresses, communicating with an executing application, accessing application memory, converting data into human-readable form, and distributed CPU architectures.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.038.html

CMU/SEI-87-TR-037
ADA188930

Prototype Real-Time Monitor: User's Manual

Van Scoy, R.; Plinta, C.; D'Ippolito, R.; Lee, K.; Rissman, M.

This report defines the user interface to the prototype real-time monitor (RTM). It defines the concepts and commands needed by a software engineer to use the RTM. In addition to defining the user interface, the report explains the steps needed to tailor the RTM to work with the users application.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.037.html

CMU/SEI-87-TR-036
ADA188929

Prototype Real-Time Monitor: Requirements

Van Scoy, R.; Plinta, C.; D'Ippolito, R.; Lee, K.; Rissman, M.

The requirements imposed by flight simulators and good software engineering practice on ADA systems force software engineers to seek new solutions to the problem of monitoring executing software. This report examines some of these requirements and, based on these requirements, defines a subset for implementation as a prototype real-time monitor (RTM).

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.036.html

CMU/SEI-87-TR-035
ADA188928

Prototype Real-Time Monitor: Executive Summary

Van Scoy, R.; Plinta, C.; D'Ippolito, R.; Lee, K.; Rissman, M.

This report summarizes the history, goals, and conclusions of the prototype real-time monitor development effort. This effort was undertaken to address two specific technical questions: 1) How can user tools find, access, and display data hidden in the bodies of ADA applications? 2) How can user tools be layered on top of ADA applications? The effort resulted in a generally usable prototype, which is documented by four other SEI reports (CMU/SEI-87-TR-036 through CMU/SEI-87-TR-039).

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.035.html

CMU/SEI-87-TR-034
ADA200605

Inertial Navigation System Simulator Program: Top-Level Design

Klein, M.; Landherr, S.

A real-time ADA application, an Inertial Navigation System (INS) simulator, is being developed by the Real-Time Embedded Systems Testbed Project as a vehicle to analyze issues regarding the use of ADA in the real-time embedded domain and to provide a context for future experimentation. The technical philosophy behind developing a real-time ADA artifact is to: (1) select a representative (e.g., strict timing demands, multiple concurrent activities, low-level I/O, error handling, interrupts, and periodic activities) real-time application; (2) use ADA tasks as the unit of concurrency for the real-time design; and (3) apply any relevant practical results being produced by the real-time scheduling research community. In particular, the INS simulator must satisfy a set of timing requirements that are similar to an INS with respect to data updating, message transmission, and message reception. This document discusses the top-level design of this application from three points of view: data flow perspective, concurrency and control perspective, and the ADA module perspective.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.036.html

CMU/SEI-87-TR-033
ADA200604

Inertial Navigation System Simulator: Behavioral Specification

Landherr, S.; Klein, M.

The Real-Time Embedded Systems Testbed Project at the SEI is specifying and developing a representative real-time application. This document augments an original set of specifications written by a Navy affiliate. The purpose of this behavioral specification is to clarify and augment the original.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.033.html

CMU/SEI-87-TR-032
ADA200612

VAXELN Experimentation: Programming a Real-Time Periodic Task Dispatcher Using VAXELN ADA 1.1

Borger, M.

The purpose of this paper is to provide the reader with some technical information and observations, ADA source code, and measurement results based on experimentation with respect to developing a real-time periodic task dispatcher in ADA. In this context, a periodically scheduled task set implies that each task in the set is executed at its own fixed frequency; a periodic task dispatcher is a software component that schedules the individual tasks at their implied runtime frequency. The results presented here are specific to a MicroVAX-II/VAXELN 2.3 target system, the VAXELN 1.1 ADA compiler, and a KWV11-C programmable real-time clock. Specifically, these results provide answers to the question: How can one achieve the effect of scheduling a set of periodic ADA tasks when the runtime frequency of some of the individual tasks is less than the clock-cycle frequency supported by an ADA runtime implementation?

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.032.html

CMU/SEI-87-TR-031
ADA200609

Annual Technical Report for ADA Embedded Systems Testbed Project

Weiderman, N.; Altman, N.; Borger, M.; Donohoe, P.; Hefley, W.; Klein, M.; Landherr, S.; Mumm, H.; Slusarz, J.

The purpose of the ADA Embedded Systems Testbed Project (now called the Real-Time Embedded Systems Project) is to investigate some of the critical issues in using ADA for real-time embedded applications, particularly the extent and quality of the runtime support facility provided by ADA implementations. The project's objective has been to generate new information about using ADA in real-time embedded systems. This information is in the form of benchmark test results, higher level experiment results, and lessons learned in designing and implementing real-time applications in ADA. This technical report provides an overview of the results produced in the first year of the project (through September 30, 1987). Details of these results are contained in other referenced technical reports.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.031.html

CMU/SEI-87-TR-030
ADA188932

Criteria for Constructing and Using an ADA Embedded System Testbed

Weiderman, N.

The purpose of this report is to list some of the criteria used in five aspects of the project: the hardware configuration, the software configuration, the real-time application, the ADA real-time experiments, and the benchmarking and instrumentation techniques. Each criterion will include a rationale. Each of the criteria listed in this report will be categorized as either essential, highly desirable, or desirable.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.030.html

CMU/SEI-87-TR-029
ADA188100

VAXELN Experimentation: Programming a Real-Time Clock and Interrupt Handling Using VAXELN ADA 1.1

Borger, M.

This report describes the results of implementing an interrupt handler totally in ADA for a MicroVAX II/VAXELN 2.3 target system, the VAXELN 1.1 ADA compiler, and a KWV11-C programmable real-time clock. It provides an overview of VAXELN interrupt handlers and the operation of the real-time clock; discusses and demonstrates the use of VAXELN kernel services to establish a link between the clock's interrupt and the starting address of an interrupt service routine; presents an ADA package of interfaces to the KWV11-C device; provides ADA source code examples demonstrating the use of this package; and presents relevant observations, recommendations, and measurement results.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.029.html

CMU/SEI-87-TR-028
ADA200608

Survey of Real-Time Performance Benchmarks for the ADA Programming Language, A

Donohoe, P.

This survey provides a summary description of some of the major ADA benchmarks currently available and an evaluation of their applicability to the Real-Time Embedded Systems Testbed Project at the SEI. The benchmarks discussed are the University of Michigan benchmarks, the ACM Performance Issues Working Group (PIWG) benchmarks, and the prototype ADA Compiler Evaluation Capability (ACEC) of the Institute for Defense Analyses (IDA).

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.028.html

CMU/SEI-87-TR-027
ADA200607

Ada Performance Benchmarks on the MicroVAX II: Summary and Results, Version 1.0

Donohoe, P.

The primary purpose of the Ada Embedded Systems Testbed (AEST) Project at the Software Engineering Institute (SEI) is to develop a solid in-house support base of hardware, software, and personnel to permit the investigation of a wide variety of issues related to software development for real-time embedded systems. Two of the most crucial issues to be investigated are the extent and quality of the facilities provided by Ada runtime support environments. The SEI support base will make assessments possible of the readiness of the Ada language and Ada tools to develop embedded systems. The benchmarking/instrumentation subgroup was formed to:

1. Collect and run available Ada benchmark programs from a variety of sources on a variety of targets.
2. Identify gaps in the coverage and fill them with new test programs.
3. Review the measurement techniques used and provide new ones if necessary.
4. Verify software timings by inspection and with specialized test instruments.

This report documents the results obtained from running Ada performance benchmarks on a DEC VAXELN MicroVAX II using the DEC VAXELN Ada compiler. The benchmarks were the University of Michigan Ada benchmarks and the ACM SIGAda Performance Issues Working Group (PIWG) Ada benchmarks (excluding the compilation tests). A description of these suites and the reasons for choosing them are given in [Donohoe87a]. The benchmarks focus largely on the execution time of specific features of the Ada language; they do not, for example, measure the efficiency or the size of the generated object code. A brief description of the benchmarks and the test environment is followed by a discussion of some problems encountered and lessons learned. The results obtained from running the entire Michigan and PIWG benchmark suites are contained in the appendices to this report. Note that the caveats discussed in the body of the report must be borne in mind when examining these results.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.027.html

CMU/SEI-87-TR-026
ADA191096

ADA for Embedded Systems: Issues and Questions

Weiderman, N.; Borger, M.; Cappellini, A.; Dart, S.; Klein, M.; Landherr, S.

This report addresses issues and questions related to the use of ADA for embedded systems applications; it contains some preliminary recommendations for compilation system implementors, application developers, program managers, and ADA policy makers. The issues and questions provide the context for the Real-Time Embedded Systems Testbed (REST) Project at the SEI, where staff members are investigating software development and performance issues for real-time embedded systems.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.026.html

CMU/SEI-87-TR-025
ADA200611

Final Evaluation of MIPS M/500 Final Report for the RISC Insertion Project

Klein, D.; Firth, R.

In response to a request from the DoD, an analysis of a Reduced Instruction Set Computer (RISC) processor, the MIPS M/500, was performed. All aspects of processor capabilities and support software were evaluated, tested, and compared to familiar Complex Instruction Set Computer (CISC) architectures. In all cases, the RISC computer and its support software performed better than a comparable CISC computer. This report provides the general and specific results of these analyses, along with the recommendation that the DoD and other government agencies seriously consider this or other RISC architectures as a highly viable and attractive alternative to the more familiar but less efficient CISC architectures.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.025.html

CMU/SEI-87-TR-024
ADA200542

Software Development

Dart, S.; Ellison, B.

“Environment” refers to the collection of hardware and software tools that a system developer uses to build software systems. As technology improves and user expectations grow, an environment’s functionality tends to change. Over the last 20 years, the set of software tools available to developers has expanded considerably. We can illustrate this change by observing some distinctions in the terminology. “Programming environment” and “software development environment” are often used synonymously, but here we make a distinction between the two.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.024.html

CMU/SEI-87-TR-023
ADA187230

Method for Assessing the Software Engineering Capability of Contractors, A

Humphrey, W.; Sweet, W.

This document provides guidelines and procedures for assessing the ability of potential DoD contractors to develop software in accordance with modern software engineering methods. It includes specific questions and a method for evaluating the results.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.023.html

CMU/SEI-87-TR-022
ADA187231

Factors Causing Unexpected Variations in ADA Benchmarks

Altman, N.

Benchmarks are often used to describe the performance of computer systems. This report considers factors that may cause ADA benchmarks to produce inaccurate results. Included are examples from the ongoing benchmarking efforts of the ADA Embedded Systems Testbed (AEST) Project using bare target computers with several ADA compilers.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.022.html

CMU/SEI-87-TR-021
ADA185697

Timing Variation in Dual Loop Benchmarks

Altman, N.; Weiderman, N.

Benchmarks that measure time values using a standard system clock often employ a dual loop design. One of the important assumptions of this design is that textually identical loop statements will take the same amount of time to execute. This assumption was tested on two bare computers with ADA test programs and has been demonstrated to be inaccurate in these specific test cases.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.021.html

CMU/SEI-87-TR-020
ADA200603

Teaching a Project-Intensive Introduction to Software Engineering

Tomayko, J.

This report is meant as a guide to the teacher of the introductory course in software engineering. It contains a case study of a course based on a large project. Other models of course organization are also discussed. Additional materials used in teaching the course and samples of student-produced documentation are also available.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.020.html

CMU/SEI-87-TR-019
ADA188926

Use of Representation Clauses and Implementation-Dependent Features in ADA: IVA. Qualitative Results for ADA/M(44), The

Meyers, C.; Cappellini, A.

This report, one in a series, provides a qualitative assessment of the support of representation clauses and implementation-dependent features in ADA provided by the ADA/M(44) compiler, Version 1.6. The evaluation questions that were presented in a previous report of this series form the basis of the qualitative assessment. A subjective evaluation of the support provided for these features is also presented.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.019.html

CMU/SEI-87-TR-018
ADA200602

Use of Representation Clauses and Implementation-Dependent Features in ADA: IIB. Experimental Procedures, The

Meyers, C.; Cappellini, A.

This report is one in a series dealing with the use of representation clauses and implementation-dependent features in ADA. The purpose of this report is to discuss detailed experimental procedures to assess compiler support. It is readily acknowledged that the domain of possible experimentation is large. To facilitate the experimentation, a methodology is proposed that relies on program generators and automated analysis tools. An example of the methodology is presented in some detail.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.018.html

CMU/SEI-87-TR-017
ADA188926

Use of Representation Clauses and Implementation-Dependent Features in ADA: IIIA. Qualitative Results for VAX ADA, The

Meyers, C.; Cappellini, A.

This report, one in a series, provides a qualitative assessment of the support of representation clauses and implementation-dependent features in ADA provided by the VAX ADA compiler, Version 1.3. The evaluation questions that were presented in a previous report of this series form the basis of the qualitative assessment. A subjective evaluation of the support provided for these features is also presented.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.017.html

CMU/SEI-87-TR-016
ADA183429

Preliminary Report on Conducting SEI-Assisted Assessments of Software Engineering

Humphrey, W.; Kitson, D.

Characterizing the state of software engineering practice within an organization is a necessary prerequisite to orderly, meaningful, and sustainable improvement of the organizations ability to produce or support cost-effective, high quality software products. The Software Engineering Institute is developing a methodology for conducting SEI-assisted assessments of software engineering capability. The assessment methodology has five phases: 1. selecting the candidate organization, 2. preparing for the assessment, 3. conducting the assessment, 4. communicating final assessment findings and action recommendations, and 5. post-assessment follow-up activities. This report describes the methodology in detail.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.016.html

CMU/SEI-87-TR-015
ADA188925

Use of Representation Clauses and Implementation-Dependent Features in ADA: IIA. Evaluation Questions, The

Meyers, C.; Cappellini, A.

This report is the second in a series on the use of representation clauses and implementation-dependent features in ADA. It is the purpose of this document to specify a set of questions relevant to the assessment of the support of representation clauses and implementation-dependent features provided by an ADA compiler. The questions identified are categorized according to functionality and address both qualitative and quantitative aspects.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.015.html

CMU/SEI-87-TR-014
ADA200601

Use of Representation Clauses and Implementation-Dependent Features in ADA: I. Overview, The

Meyers, C.; Cappellini, A.

This report, the first in a series, presents an overview of the aspects of the ADA language relating to representation clauses and implementation-dependent features. Particular emphasis is given to the use of ADA for application to packed data structures. This report is in part tutorial, and several examples from real-time, mission-critical systems are discussed in detail. A brief discussion of design guidelines for the use of representation clauses and implementation-dependent features is included.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.014.html

CMU/SEI-87-TR-013
ADA185742

Seeking the Balance Between Government and Industry Interests in Software Acquisition. Volume I. A Basis for Reconciling DoD and Industry Needs for Rights in Software

Martin, A.; Deasy, K.

The policy under which the Department of Defense (DoD) acquires rights in software and technical data has, in the past, been imbalanced in the direction of obtaining more rights than necessary to meet its needs. As noted by the Packard Commission, a more balanced policy is in the interests of both the DoD and industry. The DoD has recently adopted a new policy for acquiring rights in technical data, and is developing a separate policy for acquiring rights in software. This report offers several recommendations for achieving a balanced policy as to government funded software, privately funded software, and mixed funding software that will meet the mission needs of the DoD while enabling contractors to protect their proprietary interests, and commercialize their software products.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.013.html

CMU/SEI-87-TR-010
ADA213968

Guide to the Classification and Assessment of Software Engineering Tools, A

Firth, R.; Mosley, V.; Pethia, R.; Roberts Gold, L.; Wood, W.

Software engineering tools are computer programs that assist people in doing the work of software engineering. As understanding of the software engineering process has broadened and the need to solve problems has intensified, there has been increasing interest in using software engineering tools. Understanding what a tool does and comparing it to similar tools are difficult tasks given the diversity of functionality that exists. This report describes a tool classification technique that helps those investigating tools decide where a tool fits in the software engineering process and identify what a tool does or doesn't do. It also provides guidance to the tool evaluation process and lists specific criteria that should be considered when evaluating tools.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.010.html

CMU/SEI-87-TR-009
ADA182023

ADA Adoption Handbook

Foreman, J.; Goodenough, J.

The ADA Adoption Handbook provides program managers with information about how best to tap ADA's strengths and manage this new software technology. Although the issues are complex, they are not all unique to ADA. Indeed, many of these issues must be addressed when using any language for building sophisticated systems. The handbook addresses the advantages and risks inherent in adopting ADA. Significant emphasis has been placed on providing information and suggesting methods that will help program and project managers succeed in adopting ADA across a broad range of application domains.

The handbook focuses on the following topics: program management issues including costs and technical and program control; ADA's goals and benefits; software tools with emphasis on compiler validation and quality issues; the state of ADA technology as it relates to system engineering; the application of special purpose languages; issues related to mixing ADA with other languages; possible productivity benefits resulting from software reuse; and implications for education and training.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.009.html

CMU/SEI-87-TR-008
ADA182003

Software Engineering Education: An Interim Report from the Software Engineering Institute

Ford, G.; Gibbs, N.; Tomayko, J.

The goals and activities of the Software Engineering Institute's Education Program are described. Two curriculum recommendations are presented, one for a professional Master of Software Engineering degree program, and the other for an undergraduate project course in software engineering. Also presented is an organizational structure for software engineering curriculum content.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.008.html

CMU/SEI-87-TR-007
ADA181156

Tool Interface Technology

Newcomer, J.

This report is one of a series of survey reports. It is not intended to provide an exhaustive discussion of topics pertinent to the area of user interface technology. Rather, it is intended as an informative review of the technology surveyed. These surveys were conducted in late 1985 and early 1986.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.007.html

CMU/SEI-87-TR-006
ADA181852

User Interface Technology Survey

Feiler, P.

This report is one of a series of survey reports. It is not intended to provide an exhaustive discussion of topics pertinent to the area of distributed systems technology. Rather, it is intended as an informative review of the technology surveyed. These surveys were conducted in late 1985 and early 1986.

One of the core technology areas in which project members conducted a survey was user interface technology. This report attempts to do two things: specify an understanding of user interfaces by presenting a taxonomy that encompasses the various aspects of user interfaces, and indicate the state of the technology today by highlighting some of the major issues.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.006.html

CMU/SEI-87-TR-005
ADA181853

Distributed Systems Technology Survey

Cooper, E.

This report is one of a series of survey reports. It is not intended to provide an exhaustive discussion of topics pertinent to the area of distributed systems technology. Rather, it is intended as an informative review of the technology surveyed. These surveys were conducted in late 1985 and early 1986.

One of the core technology areas in which project members were interested is distributed systems technology. This report surveys the technical issues involved in designing distributed systems, with particular emphasis on those aspects that affect software engineering environments.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.005.html

CMU/SEI-87-TR-004
ADA182982

Software and System Warranty Issues

Druffel, L.; Wood, W.; Pethia, R.

This report addresses technical and administrative issues associated with the system warranty process, and recommends a straightforward, two-page generic system warranty clause that covers software, not in isolation, but as part of a warranted system. The report describes one approach to relieving problems of system failure, and addresses legal, technical, and administrative issues that support warranty enforcement. The goal is to ease the government's burden of proving the existence of a defect for which the warranty clause provides a remedy. The key to satisfying that goal is to develop technical tests and specifications that provide objective and demonstrable standards against which a claim for breach of warranty can be measured.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.004.html

CMU/SEI-87-TR-002
ADA178971

Effect of Software Support Needs on DoD Software Acquisition Policy: Part 1: A Framework for Analyzing Legal Issues, The

Martin, A.; Deasy, K.

This report summarizes the significant technical and managerial considerations that affect the maintenance and enhancement of software. Prior work suggested that it is often in the acquisition of intellectual property needed to maintain and enhance software that data rights disputes arise between DoD and the private sector. For this reason, an understanding of DoD's maintenance and enhancement requirements is a necessary predicate toward shaping a data rights/software acquisition policy that achieves the proper balance between the intellectual property needs of DoD and the proprietary interests of private industry. A survey of software engineering literature revealed no study that addressed this important subject. Accordingly, the Software Licensing Project undertook to examine the issue itself. Although this report discusses technical and managerial issues, it is principally intended as a guide for lawyers and policymakers who deal with, and have regulatory responsibility for, software and data rights acquisition issues.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.002.html

CMU/SEI-87-TR-001
ADA180905

Evaluation of ADA Environments

Weiderman, N.; Altman, N.; Borger, M.; Klein, M.; Landherr, S.; Smeaton, R.; D'Ippolito, R.; Kochmar, J.; Sun, A.

The full report provides a detailed description of the methodology and examples of its usage. Chapter 1 gives an extended cross-environment analysis of the results of the project. For each of five experiment groups, it compares three APSEs. The chapter provides an overview of the results of all the experiments and is written for the technical manager. Chapter 2 describes in detail the methodology used for evaluating the environments, along with some of the background information and references to previous work in environment evaluation. Chapters 3 through 8 provide detailed descriptions of the six experiment groups. Here one can find the information on particular criteria, questions, and life cycle activities that were tested for each experiment, as well as test scripts, checklists, and resulting data that were collected.

http://www.sei.cmu.edu/publications/documents/87.reports/87.tr.001.html

Curriculum Modules and Support Materials

SEI-CM-10-1.0
ADA236120

Models of Software Evolution: Life Cycle and Process

Scacchi, W. (University of Southern California)

October 1987

This module presents an introduction to models of software system evolution and their role in structuring software development. It includes a review of traditional software life-cycle models as well as software process models that have been recently proposed. It identifies three kinds of alternative models of software evolution that focus attention to either the products, production processes, or production settings as the major source of influence. It examines

how different software engineering tools and techniques can support life-cycle or process approaches. It also identifies techniques for evaluating the practical utility of a given model of software evolution for development projects in different kinds of organizational settings.

http://www.sei.cmu.edu/publications/documents/cms/cm.010.html

SEI-CM-8-1.0
ADA236362

Formal Specification of Software

Berztiss, A. (University of Pittsburgh)

October 1987

This module introduces methods for the formal specification of programs and large software systems, and reviews the domains of application of these methods. Its emphasis is on the functional properties of software. It does not deal with the specification of programming languages, the specification of user-computer interfaces, or the verification of programs. Neither does it attempt to cover the specification of distributed systems.

http://www.sei.cmu.edu/publications/documents/cms/cm.008.html

SEI-SM-8-1.0
ADA236121

Support Materials for Formal Specification of Software

Berztiss, A. (ed.) (University of Pittsburgh)

October 1987

This support materials package includes materials helpful in teaching a course on formal specification of software.

http://www.sei.cmu.edu/publications/documents/cms/sm.008.html

SEI-CM-7-1.1
ADA235924

Assurance of Software Quality

Brown, B. (Boeing Military Airplane Company)

July 1987

This module presents the underlying philosophy and associated principles and practices related to the assurance of software quality. It includes a description of the assurance activities associated with the phases of the software development life-cycle (e.g., requirements, design, test, etc.).

http://www.sei.cmu.edu/publications/documents/cms/cm.007.html

SEI-CM-6-1.1
ADA238560

Software Safety

Leveson, N. (University of California, Irvine)

July 1987

Software safety involves ensuring that software will execute within a system context without resulting in unacceptable risk. Building safety-critical software requires special procedures to be used in all phases of the software development process. This module introduces the problems involved in building such software along with the procedures that can be used to enhance the safety of the resulting software product.

http://www.sei.cmu.edu/publications/documents/cms/cm.006.html

SEI-CM-5-1.2
ADA236208

Information Protection

Cohen, F. (Lehigh University)

July 1987

This curriculum module is a broad based introduction to information protection techniques. Topics include the history and present state of cryptography, operating system protection, network protection, data base protection, physical security techniques, cost benefit tradeoffs, social issues, and current research trends. The successful student in this course will be prepared for an in-depth course in any of these topics.

http://www.sei.cmu.edu/publications/documents/cms/cm.005.html

SEI-CM-4-1.4
ADA235702

Software Configuration Management

Tomayko, J. (The Wichita State University)

July 1987

Software configuration management encompasses the disciplines and techniques of initiating, evaluating, and controlling change to software products during and after the development process. It emphasizes the importance of configuration control in managing software production.

http://www.sei.cmu.edu/publications/documents/cms/cm.004.html

1986 Reports

Technical Reports

CMU/SEI-86-TR-006
ADA178771

Heterogeneous Machine Simulator, The

Stockton, R.

The heterogeneous machine simulator is a program which attempts to simulate the proposed hardware for the heterogeneous machine at a high level, along with the low level programming abstractions which have been proposed. This will, hopefully, provide: 1) a reasonable basis for programmers to evaluate application designs in the absence of the actual machine; and 2) a testbed for designers to experiment with various reconfigurations which might be difficult to perform on the machine itself. This document presents a basic description of the system, and an example of how a simulation may be run.

http://www.sei.cmu.edu/publications/documents/86.reports/86.tr.006.html

CMU/SEI-86-TR-005
ADA200085

Summary of the SEI Workshop on Software Configuration Management

Harvey, K.

This report summarizes the discussion held during the Software Configuration Management meeting at the Software Engineering Institute in Pittsburgh on 16 July 1986.

http://www.sei.cmu.edu/publications/documents/86.reports/86.tr.005.html

CMU/SEI-86-TR-004
ADA178769

Specifying Functional and Timing Behavior for Real-Time Applications

Barbacci, M.; Wing, J.

We present a notation and a methodology for specifying the functional and timing behavior of real-time applications for a heterogeneous machine. In our methodology, we build upon well-defined, though isolated, pieces of previous work: Larch and Real-Time Logic. In our notation, we strive to keep separate the functional specification from the timing specification so that a task's functionality can be understood independent of its timing behavior. We show that while there is a clean separation of concerns between these two specifications, the semantics of both pieces as well as their combination are simple.

http://www.sei.cmu.edu/publications/documents/86.reports/86.tr.004.html

CMU/SEI-86-TR-003
ADA178975

Durra: A Task-Level Description Language Preliminary Reference Manual

Barbacci, M.

Durra is a language designed to support the development of large-grained parallel programming applications. This document is a preliminary reference manual for the syntax and semantics of the language.

http://www.sei.cmu.edu/publications/documents/86.reports/86.tr.003.html

CMU/SEI-86-TR-002
ADA182093

Proposal for a New "Rights in Software" Clause for Software Acquisitions by the Department of Defense

Samuelson, P.; Deasy, K.; Martin, A.

This report recommends three distinct regulatory strategies for addressing difficulties the DoD has been experiencing with respect to legal issues related to software acquisitions. First, the report reiterates the Software Licensing Project's earlier recommendation that the DoD adopt the proposed Federal Acquisition Regulation (FAR) data rights provisions instead of the proposed revisions to the DoD supplement to the FAR (DoD FAR SUPP). Secondly, in the event that the DoD chooses to adopt a data rights procurement policy different from that found in the data rights provisions of the proposed FAR, this report recommends that the DoD adopt a separate "Rights in Software" clause for software acquisitions, rather than continuing the present practice of handling software procurements under the "Rights in Technical Data" clause. Reasons in support of a separate software acquisition policy, as well as a beginning model "Rights in Software" clause are offered. Finally, in the event that the DoD elects to retain the procurement format presently found in the DoD FAR SUPP provisions governing software and technical data acquisitions, this report offers several concrete recommendations for changes to those regulations which should result in a procurement policy which more effectively meets the mission needs of the Defense Department.

http://www.sei.cmu.edu/publications/documents/86.reports/86.tr.002.html

CMU/SEI-86-TR-001
ADA169705

Toward a Reform of the Defense Department Software Acquisition Policy

Samuelson, P.

A series of about 120 interviews were conducted with DoD personnel and others recommended by them. This report is an organized catalog of software acquisition problems reported, along with some assessments of their seriousness.

http://www.sei.cmu.edu/publications/documents/86.reports/86.tr.001.html

Curriculum Modules and Support Materials

SEI-SM-4-1.0
ADA235511

Support Materials for Software Configuration Management

Tomayko, J. (ed.) (The Wichita State University)

September 1986

This support materials package includes materials helpful in teaching a course on configuration management.

http://www.sei.cmu.edu/publications/documents/cms/sm.004.html

Conference and Workshop Records

Many SEI conference and workshop proceedings can be purchased from Springer Verlag. Springer Verlag may be contacted at—

Springer-Verlag New York, Inc.
Service Center Secaucus
44 Hartz Way
Secaucus, NJ 07094
Phone: 1800-SPRINGER

0-387-55963-9
Software Engineering Education: SEI Conference 1992
Sledge, C. (ed.)

Proceedings of the 1992 SEI Conference on Software Engineering Education, held in San Diego, CA (Lecture Notes in Computer Science No. 640.)

0-387-54502-6
Software Engineering Education: SEI Conference 1991
Tomayko, J. (rd.)

Proceedings of the 1991 SEI Conference on Software Engineering Education, held in Pittsburgh, PA (Lecture Notes in Computer Science No. 536.)

0-387-97274-9
Software Engineering Education: SEI Conference 1990
Deimel, L. (ed.)

Proceedings of the 1990 SEI Conference on Software Engineering Education, held in Pittsburgh, PA (Lecture Notes in Computer Science, No 423.)

0-387-97090-8
Software Engineering Education: SEI Conference 1989
Gibbs, N. (ed.)

Proceedings of the 1989 SEI Conference on Software Engineering Education, held in Pittsburgh, PA (Lecture Notes in Computer Science No. 376.)

0-387-96854-7
Software Engineering Education: SEI Conference 1988
Ford, G. (ed.)

Proceedings of the 1988 SEI Conference on Software Engineering Education, held in Fairfax, VA (Lecture Notes in Computer Science No. 327.)

0-387-96840-7

Issues in Software Engineering Education: Proceedings of the 1987 SEI Conference

Fairley, R.; Freeman, P. (eds.)

Proceedings of the 1987 SEI Conference on Software Engineering Education, held in Monroeville, PA.

0-387-96469-X

Software Engineering Education: The Educational Needs of the Software Community

Gibbs, N.; Fairley, R. (Eds.)

This volume contains the extended proceedings of the 1986 Software Engineering Education Workshop, held at the SEI and sponsored by the SEI and the Wang Institute of Graduate Studies. This workshop of invited software engineering educators focuses on master's level education in software engineering, with some discussion of undergraduate and doctoral level issues.

0-7923-9361-9

Practitioner's Handbook for Real-Time Analysis, A (Guide to Rate Monotonic Analysis for Real-Time Systems)

Klein, M.; Ralya, T.; Pollak, B.; Obenza, R.; González Harbour, M.

This handbook contains a collection of quantitative methods that enable real-time system developers to understand, analyze, and predict the timing behavior of many real-time systems. The methods are practical and theoretically sound, and can be used to assess design tradeoffs and to troubleshoot system timing behavior. We call this collection of methods rate monotonic analysis (RMA).

This handbook has been created to serve as a definitive source of information and a guide for real-time developers as they analyze and design real-time systems.

In North America, available from:
Kluwer Academic Publishers Group
101 Philip Drive
Assinippi Park
Norwell, Massachusetts 02061

Everywhere else, available from:
Kluwer Academic Publishers Group
Distribution Centre
Post Office Box 322
3300 AH Dordrecht, The Netherlands

Authors

H

I

J

K

L

M

N

O

P

Q

R

Numbers

Titles