Carnegie Mellon University
Software Engineering Institute

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

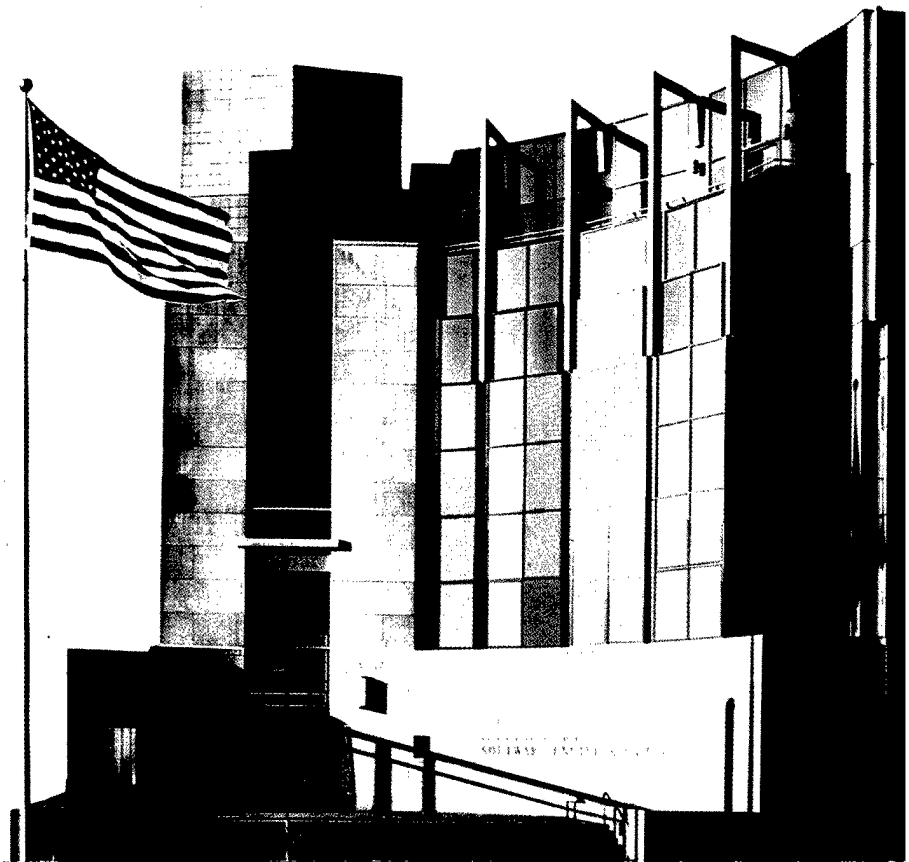
Paul C. Clements
Nelson Weiderman

*May 1998*

SR

SPECIAL REPORT
CMU/SEI-98-SR-003





Carnegie Mellon University does not discriminate and Carnegie Mellon University is required not to discriminate in admission, employment, or administration of its programs or activities on the basis of race, color, national origin, sex or handicap in violation of Title VI of the Civil Rights Act of 1964, Title IX of the Educational Amendments of 1972 and Section 504 of the Rehabilitation Act of 1973 or other federal, state, or local laws or executive orders.

In addition, Carnegie Mellon University does not discriminate in admission, employment or administration of its programs on the basis of religion, creed, ancestry, belief, age, veteran status, sexual orientation or in violation of federal, state, or local laws or executive orders. However, in the judgment of the Carnegie Mellon Human Relations Commission, the Department of Defense policy of, "Don't ask, don't tell, don't pursue," excludes openly gay, lesbian and bisexual students from receiving ROTC scholarships or serving in the military. Nevertheless, all ROTC classes at Carnegie Mellon University are available to all students.

Inquiries concerning application of these statements should be directed to the Provost, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone (412) 268-6684 or the Vice President for Enrollment, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone (412) 268-2056.

Obtain general information about Carnegie Mellon University by calling (412) 268-2000.

CMU/SEI-98-SR-003

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

Paul C. Clements
Nelson Weiderman

*May 1998*

**Product Line Systems**

Carnegie Mellon University
**Software Engineering Institute**

Pittsburgh, PA
15213-3890

This report was prepared for the

SEI Joint Program Office
HQ ESC/DIB
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

Jay Alonis, Lt Col, USAF
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense.



This work was created in the performance of Federal Government Contract Number F19628-95-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 52.227-7013.

This document is available through Asset Source for Software Engineering Technology (ASSET): 1350 Earl L. Core Road; PO Box 3305; Morgantown, West Virginia 26505 / Phone: (304) 284-9000 or toll-free in the U.S. 1-800-547-8306 / FAX: (304) 284-9001 World Wide Web: http://www.asset.com / e-mail: sei@asset.com

Copies of this document are available through the National Technical Information Service (NTIS). For information on ordering, please contact NTIS directly: National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161. Phone: (703) 487-4600.

This document is also available through the Defense Technical Information Center (DTIC). DTIC provides access to and transfer of scientific and technical information for DoD personnel, DoD contractors and potential contractors, and other U.S. Government agency personnel and their contractors. To obtain a copy, please contact DTIC directly: Defense Technical Information Center / Attn: BRR / 8725 John J. Kingman Road / Suite 0944 / Ft. Belvoir, VA 22060-6218 / Phone: (703) 767-8274 or toll-free in the U.S.: 1-800 225-3842.

# Table of Contents

# Abstract

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.

# 1 Background

The Second International Workshop on Development and Evolution of Software Architectures for Product Families dealt with topics that were germane to both the Product Line Practice and Architecture Tradeoff Analysis initiatives of the Software Engineering Institute's (SEI) Product Line Systems Program. The purpose of this report is to summarize the contents and results of the workshop and to relate them to some of the work being done at the SEI.

Many companies are looking for ways to minimize the costs of developing new products and maximize sharing and reuse of software structure and components used in a product family. As described in the conference call for papers, the primary focus of this workshop was on "methods, techniques, and tools to manage the diversity of products in a family at the level of software architecture. Topics of interest also included specification of software architecture, architecture recovery, assessment of software architecture, and other subjects related to development and evolution of software architecture for product families."

Hence, the focus was on issues of software architecture, but was oriented toward architecture issues as they apply to the creation, deployment, and evolution of families of software-intensive systems.

The workshop was attended by over 40 people, primarily from Europe. Academic institutions represented included universities in Madrid, Vienna, London, Las Palmas, Pisa, Pittsburgh, Germany, Singapore, and Sweden. Industry was present with contingents from Nokia, Philips, ABB, Lucent, Daimler Benz, VTT Electronics, and Motorola. Institutions bridging the gap included the SEI, the Fraunhofer Institute, the Information Sciences Institute, and the European Software Institute. Just under half the participants were from universities and most of the industrial participants were from corporate research departments.

This workshop was organized by the European Strategic Programme of Research into Information Technology (ESPRIT) framework IV project number 20477, ARES (Architectural Reasoning for Embedded Software). ARES is a joint project between Philips, Nokia, ABB, London Imperial College, Technical University of Vienna, and Polytechnical University of Madrid. The main objective is to enable software developers to explicitly describe, assess, and manage architectures of embedded software families. ARES was chartered to find ways to help design reliable systems with embedded software that satisfy important quality requirements, evolve gracefully, and may be built in time and on budget. ARES also addresses the problem of relating the features that differentiate the members of a product family to an archi-

tecture for that family. ARES aims to address the variance required by a product family at the architectural level and to map a feature selection to an instance of an architecture.

The general chair was Henk Obbink of Philips. Paul Clements of the SEI was the co-chair. The program coordinators were Frank van der Linden of Philips Research and Juha Kuusela of Nokia Research.

# 2 Workshop Format

The workshop produced working sessions on the following topics:

- ARES
- Examples of Architectures for Product Lines
- Architecture Description
- Architecture Analysis
- Architecture Recovery
- Process Issues

These sessions had somewhat different titles and ordering in the workshop itself. For example, Architecture Recovery was actually called Reverse Architecting. The names and order given in this report were chosen to enhance clarity and continuity.

The majority of the workshop time was spent in technical discussion sessions. Each session was represented by four to nine papers submitted by attendees (and sessions were assigned in advance). Each session was assigned a facilitator or moderator and an ARES representative who acted as a scribe and who was assigned to produce a summary of the session. A complete list of the papers appearing in the Proceedings is provided in Appendix A.

The emphasis of the workshop was on discussion, not presentation, and authors were allowed only a few slides and a few minutes to present their ideas. Each session lasted 90 minutes. The disadvantage of this format was that it was hard to present meaningful ideas and results in such a short amount of time. The advantage of the format was that the participants quickly got to the critical issues for discussion. As it turned out, there was a handful of people who had very insightful contributions and the discussions were probably more productive than a string of longer presentations would have been.

At the conclusion of the workshop, Paul Clements presented a 30-minute summary of the sessions and the issues raised. The following sections include his summary and some additional personal observations.

# 3 Workshop Summary

## 3.1 ARES Session

First, ARES representatives spoke about current ARES work in each of the three areas of ARES inquiry – reverse architecting, description, and analysis. ARES work revolves around running case studies with tractable but real systems (such as the French train scheduling system). Most case studies are winding down, as the ARES project itself is coming to a close this year. The result will be a book detailing the case studies and their results, with a publication target of late 1998.

Alexander Ran of Nokia, the ARES project architect, introduced the ARES work by providing the motivation and overview for the research. The underlying motivation was management complaints that there was no manageable way to make small changes to requirements without generating a large amount of work to implement the changes. His summary of the ARES work was that "nothing (i.e., the research results) seemed to work" for large complex industrial-strength systems, but some things worked sometimes with various types of adjustments and accommodations. In addition to the train safety application there were three others —digital switches, mobile phones, and embedded television sets.

The group working on architecture recovery from the Technical University of Vienna started with a reengineering model for going from procedural to object-oriented architectures and were surprised to find that in working with real systems that there was no "the architecture." They found challenges in integration of software views and integration of human knowledge. The group working on description from London Imperial College used an architecture description language (ADL) called Darwin as a starting point. They found that Darwin was not suitable for the embedded systems and it evolved to a new ADL called Koala. The group working on analysis from the Polytechnical University of Madrid used various tools including the Software Architecture Analysis Method (SAAM), scenarios, checklists, metrics, prototyping, and Rate Monotonic Analysis (RMA). They had some success applying analysis technology to instances, but, with the exception of RMA, they did not address how the technology applied to product lines.

In summary, Ran said that all issues remained open and that the major problem is managing information about software from various sources. There is a need for creation, management, and use of a software information base. He illustrated the need for multiple views of software systems and opined that most often a single, non-representative view is adopted. Consequently, the view is not useful and not used. His overall message, as a practitioner, was that we

must be satisfied with slow, incremental change and that for even the simplest of tasks, we must apply architectural concepts in the context of real-world, complex, industrial applications.

## 3.2 Examples of Architectures for Product Lines

The papers in this session presented sample architectures for product lines, but there was little insight as to how these architectures were created. The presentation of results was not particularly valuable, but part of the problem was the time restriction—it is difficult to gain insight from a 5-minute presentation of an architecture. But it should be noted that at the end of the workshop, participants clamored for even more examples. However, they wanted examples of product lines, not just product line architectures—in short, they were asking for SEI-like product line case studies.

More interesting were the economic issues that were raised. Economic models, it was felt, were talked about quite a bit but not used as justification to launch a product line effort. Rather, appeals on intuitive and intellectual grounds tended to convince management to proceed. Major goals cited for product lines include

- reduced time to market
- ability to produce products with fewer people
- ability to utilize outside development (whether sub-contracted or commercial off-the-shelf (COTS))
- ability to utilize previous development (legacy systems)
- "product alignment;" for instance, the ability to certify a set of safety-critical systems all at once, or the ability to have products exhibit the same look and feel

Note the absence of "lower cost" in this list, which largely reflects the motivations we have heard at SEI workshops. (The fewer-people goal is more about the non-availability of skilled people for hire at any price, than about decreasing the cost of production.) In contrast, a group developing a medical product line cited lower cost as a major driver and explained that management was hoping for "well more than 10%" yearly in cost savings.

The highest risk, it was felt, was being able to meet all the different clients' needs with systems that were versions of each other produced from a single common asset base.

Organizational structure was touched on, but not explored. One speaker argued strongly for a separate unit to create the core asset base, explaining that when time and budget got squeezed, a single product manager would resort to building systems one at a time. During the break,

however, some participants countered with the argument that emerged from the Software Engineering Institute's second Product Line Workshop: An architecture/asset group separate from product groups will produce beauty, but not profit.

Open questions raised by the session included

- What can we discover about the architecture creation process?
- Why aren't economic models used? Are they worth pursuing?
- What are the arguments for and against a separate organizational structure for core asset creation and maintenance?

## 3.3 Architecture Description

Architecture description was seen by some of the participants as an intellectual tar pit. The architecture description language (ADL) researchers, at least in academia, have often failed to identify the purpose, the goals, or (most importantly) the audiences ADLs are intended to serve. In short, they have lost track of the practical implications of much of the work. This session shed little light on these high-level issues.

Nevertheless, discussion in this session did revolve around the following important issues, even if no resolution was suggested. For this session, the list of issues discussed is the same as the list of open issues raised:

- What information should an ADL capture?
- For whom should it be captured? That is, are ADLs only for the architect? Who else uses the architecture description rendered in an ADL?
- How do (should) ADLs handle variability? What kind of variability is present in a product line that is not present in the architectural description of a conventional system?
- Is there a "grand unified view" of architecture from which other views derive? If there is, is it useful to pursue it, or is it the case that we want the derived views, and a unified view (if it exists) is merely an intellectual curiosity?
- Where do ADLs fit in the process of creating and using an architecture? Architects almost always start out by scribbling on the back of a napkin; eventually they produce something that can be given to component builders, integrators, etc. These early and late representations of the architecture are vastly different. Where can an ADL help?

This session could have been less oriented towards ADLs as the vehicle for representation, but in many circles these days mentioning "architecture description" automatically conjures up ADLs.

## 3.4 Architecture Analysis

In this session, the workshop participants tried to draw some conclusions about architecture analysis. Discussion centered around the goals for analysis and the stakeholders for those goals. The group identified these analyses or analysis issues specific to the product line context:

- the variations encompassed in the product line (and in particular, by the generic assets, especially the architecture)
- time/cost of producing an instance
- any analysis that can be performed on the generic assets that also applies to all derived instances. For example, any performance analysis that can be applied to the generic architecture is multiply valuable if it automatically carries over to product instances that use that architecture.
- conformance of specific instances to the generic view. That is, it is important to assure that a product instance satisfies all of the constraints imposed on it by the generic architecture to insure that assertions made about the generic also apply to the specific.

The stakeholder discussion produced the following list of stakeholders specific to a product line environment:

- product line architect
- builder of generic (core) assets
- builder of product from generic assets
- product line maintainer
- marketer / funder of the product line

Towards the end of this session, representatives of the major non-academic organizations present were asked to explain their organizations' approach to architecture evaluation. The moderator started by explaining the Software Architecture Analysis Method (SAAM) and mentioning the transition to the Architecture Tradeoff Analysis Method (ATAM). David Weiss of Lucent explained their Software Architecture Review Board (SARB) process, with which we were familiar. Then the representatives of Nokia, Philips, and ABB were asked about their analysis processes. The latter three had processes in place, but it seemed that each was a variation on the standard theme of unstructured reviews.

Open issues raised by this session were

- Is there a canonical list of stakeholders and their goals for analysis? If so, what is it?
- What are the costs and qualitative/quantitative benefits of analysis?

## 3.5 Architecture Recovery

Examples of architecture recovery tools were discussed, and Dali[1] was raised as the lone example of tools that incorporate input from human experts. There seemed to be near consensus that it was necessary to use experts (system, domain) to guide the recovery process; without them, it is largely a hopeless and futile task.

As Mehdi Jazayeri put it, there are three sources of information for architecture recovery:

1. source code, which is authoritative and dependable but by no means contains all of the information
2. documentation, which is undependable, incomplete, and informal
3. human experts, who are dependable, but biased

Messy though it is, architecture recovery was thought to be an essential part of product line development: The assertion was made, and not challenged, that the great majority of product lines are built from existing assets.

Issues left open included

- Exactly what information should recovery seek to target?
- What exactly is the process for building product lines with recovered assets?

In the ARES overview presentation on this topic, there were many reverse engineering tools mentioned, including Refine/C, Imagix 4D, Rigi, and SNIFF+. It was noted that these all look similar on paper, but are different in actual operation and they do not interoperate. SEI has efforts underway in the reengineering effort in the Product Lines System Program to address the issue of integrating tools such as these. This integrating middleware is called CORUM (Common Object-based Reengineering Unified Model).

1. Dali is a tool built at the Software Engineering Institute for extracting architectural information from an implemented system. It combines off-the-shelf tools such as syntactic analyzers and profilers with a database to provide the capability for human-guided architectural analysis.

## 3.6 Process Issues

This final session served as a grab bag for papers not allocated elsewhere. The moderator tried to elicit hard data about how well organizations' processes were working (or not working), but little was forthcoming. The CelsiusTech data was noted, as well as some of the productivity improvements that were reported in the Software Engineering Institute's second Product Line Practice Workshop (no attributions were made since the report was not published at the time of the workshop).

An underlying theme of this session was the importance of stakeholder involvement, but no firm conclusions were recorded. A short discussion ensued about the role of academia in investigating product line issues: Are they fulfilling their mission by providing only technology with little concern for the organizational and enterprise issues we know to predominate in product line production? Can academics practice information hiding and ignore many of the broader issues? Can the technologists not team with business schools in their institutions to better serve the practicing community? As one might expect, this issue was not settled.

Accordingly, the open issues from this session were:

- Where is the data about improvement?
- What is the role of academia?
- What are the ways to appropriately handle variation in the process?

## 3.7 Conclusions

The impromptu summary spoke to the open issues from each session that are highlighted above. The following themes permeated all of the sessions:

- **recognition of specific goals at each step, and orienting the work (creation, description, analysis, recovery, process) towards the fulfillment of those goals:** We are not interested in architecture manipulation for its own sake, but only for the furthering of enterprise aims.
- **recognition of the importance of stakeholders:** Similar to the first point, this makes it clear that architects answer to stakeholders, and a beautiful but unprofitable architecture is in demand by no one.
- **emphasis of people over technology:** At this workshop, at least, people issues tended to dominate the presentation of exotic new technologies. (During the response session, technology was roundly defended as essential. But for whatever reason, it was not the main focus of this gathering.)

- **recognition of business and organizational concerns:** This is an instantiation of the first two points. In fact, there was agreement to devote time to these issues at any future workshop.
- **recognition of contributions from other fields:** More than one speaker pointed out that other engineering disciplines have been building product families for generations. Is it not possible that we can learn from them?

# 4 Conclusions

This was the second such workshop sponsored by ARES, and since ARES is concluding, they will not be in a position to sponsor a third workshop. The general chair asked for any expressions of interest for continuing the workshop under some other auspices. Some spoke strongly in favor, saying that interest in product line production is clearly on the rise, and we have just scratched the surface of interesting and compelling technical and organizational issues. Others also expressed interest, but were not as enthusiastic. Nevertheless, there is an opportunity to continue and enhance the work of this community, engage a broader audience, sharpen its focus to produce more tangible results, and shift the orientation to more general product line issues instead of just focusing on architectural issues.

# 5 More Information

There are tentative plans to have Springer-Verlag publish the proceedings in their *Lecture Notes in Computer Science* series. Of particular interest is the paper about stakeholders for product lines by Dolan, Weterings, and Wortman. As noted earlier, there will be a book published in late 1998 describing the results of the entire ARES project. Also, there are Web sites describing ARES research at

- http://hpv17.infosys.tuwien.ac.at/ARES/ (Vienna)
- http://www.dit.upm.es/~ares/ (Madrid)
- http://www-dse.doc.ic.ac.uk/ (London)

# Appendix A Contents of the Proceedings

ARES Session: Alexander Ran, Mehdi Jazayeri, Jeff Magee, Juan Antonio de la Puente.

**Session 1: Example Architectures**

1. G. Cysewski, T. Gromadzki, H. Lyskawa, M. Piechowka, S. Szejko, W.E. Kozlowski, O. Vahamaki, *Reusable Framework for Telecontrol Protocols*
2. Eila Niemela, Harri Perunka, Tomi Korpipaa, *A Software Bus as a Platform for a Family of Distibuted Embedded System Products*
3. Peter Kolb, Beat Huber, *A Three-Tier Design Approach for a Family of Large AC Drive Control Systems*
4. Christopher Ganz, Michael Layes, Modular turbine control software: *A Software Architecture for the ABB Gas Turbine Family Control System*
5. Andreas Roesel, Michael Wilcke, *Evolution of an Object-Oriented Framework in an Industrial Setting*

**Session 2: Description**

1. Dewayne E. Perry, *Generic Architecture Descriptions for Product Lines*
2. Nat Pryce, Steve Crane, *A Model of Interaction in Concurrent and Distributed Systems*
3. Peter van de Hamer, Frank van der Linden, Alison Saunders, Henk te Sligte, *An Integral Hierarchy and Diversity Model for Describing Product Family Architecture*
4. Rob van Ommering, *Koala, a Component Model for Consumer Electronics Product Software*
5. Jan Bosch, *Composition of Product Architectures*

**Session 3: Reverse Architecting**

1. Wolfgang Eixelsberger, Manfred Kalan, Michaela Ogris, Håkon Beckman, Brendt Bellay, Harald Gall, *Recovery of Architectural Structure: A Case Study*
2. Roland Knor, Georg Trausmuth, Johannes Weidl, *Reengineering C/C++ Source Code by Transforming State Machines*
3. Nabor C. Mendonca, Jeff Kramer, *An Experiment in Distributed Software Architecture Recovery*
4. Berndt Bellay, Harald Gall, *Reverse Engineering to Recover and Describe a System's Architecture*

**Session 4: Analysis**

1. Richard Bechtold, *Diagnostic Software Architectures*
2. Juan C. Duenas, William L. de Oliveira, Juan A. de la Puente, *Software Architecture Evaluation Model*
3. Robert Balzer, *An Architectural Infrastructure for Product Families*
4. Alejandro Alonso, Marisol García-Valls, Juan A. de la Puente, *Assessment of Timing Properties of Family Products*

**Session 5: Process**

1. Tom Dolan, *Stakeholders in Software-System Family Architectures*
2. Yu Chye Cheon, Akkihebbal L. Ananda, Stan Jarzabek, *Handling Variant Requirements in Software Architectures for Product Families*
3. Lothar Baum, Lars Geuer, Georg Molter, *Architecture-Centric Software Development Based on Extended Design Spaces*
4. Jacques Meekel, Thomas B. Horton, Charlie Mellone, *Architecting for Domain Variability*
5. David M. Weiss, Commonality Analysis: *A Systematic Process for Defining Families*
6. Anssi Karhinen, Juha Kuusela, *Structuring Design Decisions for Evolution*
7. Nelson Weiderman, John Bergey, Dennis Smith, Scott Tilley, *Can Legacy Systems Beget Product Lines?*
8. William Scherlis, *Structural views, Structural Evolution, and Product Families*
9. Jean-Marc DeBaud, Jean-Francois Girard, *The Relation Between the Product Line Development Entry Points and Reengineering*

10. Ulf Cederling, Bengt Lennartsson, *The Architectural Platform of a System Family in Evolution*

11. Jeroen Brouwer, Ad Jurriens, Henk van Kessel, Alef Schippers, *Product Family and Reuse in Separate Market Driven Profit Centers*

12. Sergio Bandinelli, *ERW'97 session report: Reuse Adoption Experiences Across a Large Corporation*

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*



| 1. AGENCY USE ONLY (leave blank) | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | May 1998 | Final |

**4. TITLE AND SUBTITLE**

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

**5. FUNDING NUMBERS**

C — F19628-95-C-0003

**6. AUTHOR(S)**

Clements, P. and Weiderman, N.

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

**8. PERFORMING ORGANIZATION REPORT NUMBER**

CMU/SEI-98-SR-003

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

HQ ESC/DIB
5 Eglin Street
Hanscom AFB, MA 01731-2116

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

**11. SUPPLEMENTARY NOTES**

**12.a DISTRIBUTION/AVAILABILITY STATEMENT**

Unclassified/Unlimited, DTIC, NTIS

**12.b DISTRIBUTION CODE**

**13. ABSTRACT** (maximum 200 words)

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.

**14. SUBJECT TERMS**

architecture analysis, architecture description, architecture process, architecture recovery, ARES, ARES workshop, product line, software architecture

**15. NUMBER OF PAGES**

20

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | UL |