This material is being posted by Carnegie Mellon University's Software Engineering Institute (SEI) on this Web site as a community service.

The First Software Product Line Conference (SPLC1) was held August 28-31, 2000, Denver, Colorado.

The Proceedings of the SPLC1 were published by Kluwer as part of the Kluwer International Series in Engineering and Computer Science:

Software Product Lines
Experience and Research Directions
Edited by **Patrick Donohoe**

Contact Kluwer to obtain a copy of the proceedings.

The SEI held the first Software Product Line Conference in Denver, CO the week of Aug 28-31. We had hoped for at least 100 participants. We had 185.

The participants came from North America (US and Canada), Europe (8 countries represented), Asia, Africa, and Australia. Most were from commercial organizations, but academia and government (especially through government contractors) were well represented. Representatives from the software product line leaders were all there (HP, Nokia, Philips, Bosch, Lucent, Avaya, Cummins Engines, Motorola, Ericsson, Thomson, General Motors, etc.).

The conference program included ten tutorials, seven workshops, a keynote presentation, two panels, twenty-seven technical paper presentations (fifty-nine papers were submitted), an event we called the Software Product Line Hall of Fame at which we had the participants nominate the software product line elite (A7 Avionics, CelsiusTech SS2000, HP Owen Printer Product Line, and Nokia mobile cell phones got inducted), a reception, and BOF sessions. All events were well attended.

The SEI was certainly not the first to discover or to succeed with software product lines. There were many innovators who paved the way. Though there are still lots of problems and hence areas in which to mature software product line practices, there is now a critical mass of influential players who recognize software product lines as an approach to engineering software systems whose time has come. There is now an official international software product line community.

The SEI will be hosting SPLC2 in 2002. We are currently planning on a format similar to SPLC1: workshops, tutorials, technical papers, panels, product line Hall of Fame, and a keynote speaker.

SPLC1 Summary

The following information summaries the SPLC1. The SPLC1 workshop summaries are available to the product-line community.

Conference Committee

Linda M. Northrop	Paul Clements	Alexander Ran
Software Engineering Institute	*Software Engineering Institute*	*Nokia*
SPLC1 Chair	*SPLC1 Program Co-Chair*	*SPLC1 Program Co-Chair*

Program Committee

- Sergio Bandinelli, *European Software Institute*
- Len Bass, *Software Engineering Institute*
- Don Batory, *University of Texas at Austin*
- Jan Bosch, *University of Karlskrona/Ronneby*
- Grady Campbell, *Prosperity Heights Software*
- TW Cook, *MCC*
- Mike Grier, *Raytheon*
- Emil Jandourek, *Hewlett-Packard*
- Mehdi Jazayeri, *University of Vienna*
- Jean Jourdan, *Thomson-CSF*
- Philippe Kruchten, *Rational Software Canada*
- Robert Nord, *Siemens*, USA
- Henk Obbink, *Philips*
- David Sharp, *Boeing*
- Karma Sherif, *Temple University*
- Theo von Bomhard, *Robert Bosch Corp*

- Dave Weiss, *Lucent Bell Labs*

Conference Contacts

Paul Clements	Gary Chastek	Felix Bachmann
Software Engineering Institute	*Software Engineering Institute*	*Robert Bosch Corp.*
SPLC1 Program Co-Chair & Panel Chair	*Tutorial Chair*	*Workshop Chair*

The SEI is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University.

The First Software Product Line Conference

Workshop Summaries

Workshops provide a valuable opportunity for small communities of people with diverse perspectives to engage in rich discussions about a topic of common interest. Workshops can focus on research or applied topics. Innovative, controversial, or highly-practical topics are particularly suitable for workshops.

The following information summarizes the SPLC1 workshops.

- Combining Architecture, Asset Management, and Culture to Successfully Develop Product Lines
- Product Line Architecture
- Architecture Reconstruction and Product Lines
- Measurement Issues for Software Product Line Engineering
- Generative Techniques for Product Lines
- Product Lines for Command-and-Control Ground Systems
- Embedded System Product Lines

The First Software Product Line Conference

Technical Papers (Authors and Abstracts)

The following information provides the title, author(s), and abstract of each of the 27 technical papers accepted for The First Software Product Line Conference. Papers are alphabetically listed based on first author.

CoPAM: A Component-Oriented Platform Architecting Method Family for Product Family Engineering

Pierre America[1], Henk Obbink[1], Rob van Ommering[1], and Frank van der Linden[2]
[1]Philips Research, Prof. Holstlaan 5656 AA, Eindhoven, The Netherlands
[2]Philips Medical Systems, Veenpluis 4–6, 5684 PC Best, The Netherlands
{pierre.america, henk.obbink, rob.van.ommering, frank.van.der.linden}@philips.com

Abstract: In this paper we describe a family of methods that enable the development of product family architectures. These methods have in common that they offer support in developing a family of software-intensive products on the basis of one or more common platforms, that they use component technology to build the platforms and the products, and that they use several well-defined software development processes. The methods differ in many important decisions regarding processes and architecture, which are tuned carefully to the business and organizational context. In this way the method family establishes a significant synergy in the development of several product families in a large and diverse industrial company.

Domain Engineered Configuration Control

Mark Ardis, Peter Dudak, Liz Dor, Wen-jenq Leu, Lloyd Nakatani, Bob Olsen, and Paul Pontrelli
Lucent Technologies
{maa, pddudak, dor, wleu, lhn, rgo, pontrelli}@lucent.com

Abstract: This paper describes the experiences of a small team of domain experts in re-engineering the configuration control software for the 5ESS® switch. The project consisted of three main phases: discovery, design, and deployment. During the discovery phase the team conducted a domain analysis of configuration control software. In the design phase the team created domain-specific languages and supporting tools. The deployment phase consisted of trial use and modification in response to feedback from users. Lessons learned during each of these phases are summarized.

Component-Based Product Line Development: The KobrA Approach

Colin Atkinson, Joachim Bayer, and Dirk Muthig
Fraunhofer Institute for Experimental Software Engineering (IESE), Sauerwiesen 6, D-67661 Kaiserslautern, Germany
{atkinson, bayer, muthig}@iese.fhg.de

Abstract: The product line and component-based approaches to software engineering both hold the potential to significantly increase the level of reuse in industrial software development and maintenance. They also have complementary strengths, since they address the problem of reuse at opposite ends of the granularity spectrum—product line development essentially supports "reuse in the large" while component based development supports "reuse in the small." This paper describes a method, KobrA, which cleanly integrates the two paradigms into a systematic, unified approach to software development and maintenance. Key synergies resulting from this integration include support for the rapid and flexible instantiation of system variants, and the provision of methodological support for component-based framework development.

Object-Oriented Frameworks and Product Lines

Don Batory[1], Rich Cardone[1], and Yannis Smaragdakis[2]
[1]Department of Computer Sciences, The University of Texas, Austin, Texas 78712
[2]College of Computing, Georgia Institute of Technology, Atlanta, Georgia 30332
{batory, richcar}@cs.utexas.edu, yannis@cc.gatech.edu

Abstract: Frameworks are a common object-oriented code-structuring technique that is used in application product lines. A *framework* is a set of abstract classes that embody an abstract design; a *framework instance* is a set of concrete classes that subclass abstract classes to provide an executable subsystem. Frameworks are designed for reuse: abstract classes encapsulate common code and concrete classes encapsulate instance-specific code. Unfortunately, this delineation of reusable vs. instance-specific code is problematic. Concrete classes of different framework instances can have much in common and there can be variations in abstract classes, all of which lead to unnecessary code replication. In this paper, we show how to overcome these limitations by decomposing frameworks and framework instances into primitive and reusable components. Doing so reduces code replication and creates a component-based product line of frameworks and framework instances.

Model-Based Requirements Engineering for Product Lines

Günter Böckle
Siemens AG, Corporate Technology, D-81730 Munich, Germany
Guenter.Boeckle@mchp.siemens.de

Abstract: Product line engineering is a major approach for developing products with many variants, or for a collection of products sharing a common set of reusable assets where a short time to market is needed or where the reuse of tested assets brings significant savings in development time and product quality. Requirements engineering is an important part of software and systems engineering but it needs some specific support for its application in product line engineering. Tracing differences of the requirements for different products of the family to different architecture instantiations and different component variants and to the corresponding design decisions on a requirements model can reduce development time significantly. Requirements engineering is used in both domain engineering and application engineering. A good requirements model can support major parts of both, such as commonality analysis and tracing of connections between requirements, architecture, components, and tests. This paper presents our approach for a requirements engineering model, as well as showing how working groups can work in parallel on products of the product family.

The SPLIT Method
Building Product Lines for Software Intensive Systems

Michel Coriat[1], Jean Jourdan[1], and Fabien Boisbourdin[2]
LCAT (Alcatel / Thomson-CSF Common Research Laboratory)
[1]Thomson-CSF / LCR
[2]Alcatel / CRC

Domaine de Corbeville - 91404 Orsay Cedex - France
{michel.coriat, jean.jourdan, fabien.boisbourdin}@lcr.thomson-csf.com

Abstract: This paper presents SPLIT (Software Product Line Integrated Technology), an experimental method that helps Thomson-CSF and Alcatel to define and build product lines. SPLIT is organized as a global framework to help engineer product lines of software intensive systems. In this context the paper focuses on three themes: product line requirements (Cloud), product line architecture (Daisy), and product line process (Wheel). Although SPLIT proposes models independently of the notation used for their description, in this paper we have illustrated the approach by the use of the unified modeling language (UML).

Cummins' Experience in Developing a Software Product Line Architecture for Real-time Embedded Diesel Engine Controls

James C. Dager
Cummins Engine Company
Jim.C.Dager@Cummins.com

Abstract: Over six years ago, Cummins Engine Company established a software product line for its real-time embedded diesel engine controls. The engine controls product line requirements domain is very complex. As the world's largest manufacturer of diesel engines over 200 horsepower, Cummins makes engines for a large, world-wide variety of applications, customers, original equipment manufacturers (OEM's), engine sizes, and engine configurations. Cummins' software product line program slashed development costs and cycle time across these highly varying products, and resulted in many successful product launches over the past four years. Over these years, Cummins has gained a great deal of experience and has crossed many hurdles in establishing its product line architecture. This paper describes Cummins' experiences developing, documenting and maintaining a software product line architecture. The paper focuses on the development process and covers topics such as domain analysis, architectural views, and use of Software Engineering Institute (SEI) practices and literature to guide Cummins' development approach. Also described are the architecture development process, the organizational challenges faced, and the lessons learned over this 6-year development window.

Freeing Product Line Architectures from Execution Dependencies

Bryan S. Doerr and David Sharp
The Boeing Company, St. Louis, MO
{bryan.s.doerr, david.sharp}@boeing.com

Abstract: Product line software architectures and supporting components are the focus of an increasing number of software organizations attempting to reduce software costs. One essential attribute of a product line architecture is that it effectively isolate the logical, or static, aspects of the application from any product-specific variations in the physical architecture, or execution environment. A primary element of this isolation is hardware and low-level software (e.g., operating system) independence. The focus of this paper is on the various ways physical architecture attributes can be designed for flexibility without introducing volatility into the application architecture.

Value-Based Software Engineering (VBSE)
A Value-Driven Approach to Product-Line Engineering

Stuart R. Faulk[1], Robert R. Harmon[2], and David M. Raffo[2]
[1]University of Oregon
[2]Portland State University
faulk@cs.uoregon.edu, {roberth, davidr}@sba.pdx.edu

Abstract: We consider a set of programs a family when it pays to look at their common aspects before looking at their differences. For commercial software developers the implications are twofold: First, making rational decisions about product-line processes and products requires the ability to answer the question: "Does it pay?" Second, whether or not something pays is ultimately a business (rather than software engineering) question. In short, making sound software engineering decisions requires understanding the business implications of those decisions, and vice versa. This paper describes work in progress to develop a product-line process model and common value metric that adequately link product value drivers (what it pays) with the software engineering decisions that affect those drivers. In it, we describe a systematic approach to quantifying the return for both product and process improvements based on common software engineering principles and a common value metric, *customer value*.

Implementing Product-Line Features by Composing Component Aspects

Martin L. Griss
Hewlett-Packard Company, Laboratories, Palo Alto, CA, USA
griss@hpl.hp.com

Abstract: Emerging e-commerce systems are highly dynamic and require even more flexibility and reduced time to market than that traditionally provided by product-line development and component-based software engineering (CBSE). In this paper we describe our work on agent-based, product-line CBSE for flexible e-commerce systems. We are integrating several technologies for product-line

analysis and component design, implementation, and customization to create a basis for systematic product-line development. Largely independent work on reuse ("domain analysis") and object-oriented technology ("design and code") has matured to the degree that integration of the techniques promises a coherent approach. We outline a practical development process that structures a set of common and variable features supporting a product line, to produce reusable elements ("aspects"). These aspects can be combined into customized components and frameworks to support the product line.

Applying Feature Models in Industrial Settings

Andreas Hein, Michael Schlick, and Renato Vinga-Martins
Robert Bosch GmbH, Corporate Research and Development – FV/SLD
P.O. Box 90 01 69, D-60441 Frankfurt am Main – Germany
{andreas.hein1,michael.schlick}@de.bosch.com

Abstract: A software product line is a collection of products sharing a common set of features that address the specific needs of a given business. The PRAISE project, partly funded by the European Commission under ESPRIT contract 28651 and pursued by Thomson-CSF/LCR (France), Robert Bosch GmbH (Germany), and the European Software Institute (Spain), is currently investigating product-line realization and its assessment in industrial settings. A part of the project is dedicated to the validation and consolidation of proposed product-line technologies in real-world industrial experiments. This paper presents the first experimental results obtained by Bosch. The Bosch experiment is located in the Car Periphery Supervision (CPS) domain. One focus has been on feasibility of variability modelling with feature-oriented domain analysis (FODA. The experiment has shown that the FODA model does not provide the necessary expressiveness to represent the different types of crosslinks that are necessary to describe the domain. This paper presents an extension to overcome this shortcoming.

Developing Engineered Product Support Applications

H. James Hoover[1], Tony Olekshy[2], Garry Froehlich[1], and Paul Sorenson[1]
[1]Dept. of Computing Science, University of Alberta, Edmonton, Alberta
[2]Avra Software Lab, Inc., Edmonton, Alberta.
{hoover, garry, sorenson}@cs.ualberta.ca, olekshy@avrasoft.com

Abstract: Product line developers have two primary problems to solve: keeping their applications relevant to the associated manufactured product line, and evolving their applications to exploit new information technologies. This paper reports on our experience in developing commercial applications for engineered product manufacturers. High quality object-oriented frameworks with good factoring into services are crucial to addressing both of these issues. We have also learned that it is not sufficient to

develop frameworks that address only the development of the application. They must also support other parts of the process: from production of documentation, through help desk integration, to defect tracking and resolution. This work also contains a catalogue of best practices and advanced features that we believe will be valuable to other builders of engineering tools.

Aspect-Oriented Analysis for Product Line Architecture

Tomoji Kishi and Natsuko Noda
Software Design Laboratories, NEC Corporation,Tokyo, Japan
{kishi, n-noda}@ccs.mt.nec.co.jp

Abstract: In designing a product-line architecture (PLA), it is important to analyze common and variable requirements in a product family. These requirements must be analyzed not only from the functional aspect but also from aspects related to quality attributes such as performance and reliability. For example, if two products are required to attain different levels of performance, architectures for these products may be different even if they provide the same functionality. In this paper, we propose an aspect-oriented analysis method for PLA design in which we analyze product requirements from each aspect separately. In the method, we identify important factors for each quality attribute, and characterize the services in terms of the factors. Based on the characterization, we separate requirements related to each quality attribute from the original requirements. Using the method, we can examine the architecture styles required for a PLA from each aspect, which can make PLA design easier.

Domain-Oriented Engineering of Elevator Control Software *A Product Line Practice*

Kwanwoo Lee[1], Kyo C. Kang[1], Eunman Koh[2], Wonsuk Chae[1], Bokyoung Kim[1], and Byoung Wook Choi[3]
[1]Department of Computer Science and Engineering, Pohang University of Science and Technology (POSTECH) San 31 Pohang Kyoungbuk 790-784, Korea
[2]Department of Computer and Communications Engineering, Graduate School for Information Technology, POSTECH. Also affiliated with LG Industrial Systems Co., Ltd.
[3]Division of Mechanical and Control Engineering, Sunmoon University; work also done at LG Industrial Systems Co., Ltd.
{kwlee, kck, emkoh, sein92, bogus}@postech.ac.kr, bwchoi@mail.lgis.co.kr

Abstract: Development and maintenance of embedded control software has been a difficult challenge for the manufacturing industry because of the diversity of customers' needs, rapidly changing market

requirements, and the quick response of market competition. In order to retain a market share in a competitive market, embedded control software must be designed to respond quickly to both the customer and the market, and be of high quality. LG Industrial Systems Co. Ltd. (LGIS), one of Korea's leading suppliers of elevator control systems, has also been faced with the same difficulty in the development and maintenance of elevator control software (ECS). In order to help LGIS improve the productivity and maintainability as well as the quality of its ECS, we have adopted a domain-oriented approach for reuse, and verification and validation technology for improving software quality. We have found that we can reduce maintenance costs drastically as we have developed the software by utilizing reusable and adaptable components that can easily accommodate contextual as well as requirement changes, and have verified and validated ECS in the early phase of development.

A Computing Model of Product Lines for Distributed Processing Systems, its Product Sets, and its Applications

Yoshitomi Morisawa
Nihon Unisys, Ltd., Tokyo, Japan; Graduate School of Information Science, Nara Institute of Science and Technology, Nara, Japan
Yoshitomi.Morisawa@unisys.co.jp

Abstract: When implementing an application system in a distributed computing environment, several architectural questions arise, such as how and where computing resources are distributed, and how the communication among computing resources should be implemented. To simplify the process of making these choices, we have developed a distributed computing model. The model classifies product lines for distributed processing systems into seven categories based on the location of data storage and the style of processing between client and server. This paper describes our model and its uses in selecting a product set in a kernel reference model of the Open Solution Framework and in planning the infrastructure of a new system for one of our customers.

Two Novel Concepts for Systematic Product Line Development

Alessandro Pasetti and Wolfgang Pree
Faculty of Computer Science, University of Constance, D-78457, Constance, Germany
pasetti@fmi.uni-konstanz.de, pree@acm.org

Abstract: Framelets and implementation cases are new concepts to manage the complexity of product line development. Framelets are "small product lines" that address, as self-standing units, specific problems within the product line. A product line is obtained as a combination of framelets. Framelets simplify the development and extension of product lines, and make their integration with other product

lines and with other software simpler. Implementation cases are introduced as ways to continuously monitor the adequacy of an evolving product line design. They describe an aspect of the product line instantiation process by specifying how an architectural feature for an application can be implemented using the constructs offered by the product line. This paper discusses the two concepts in the context of the design of a product line of on-board satellite software. Heuristics for defining framelets and implementation cases derived from the experience are also discussed.

An Interface Based Platform Approach

B.J. Pronk
Philips Medical Systems
ben.pronk@philips.com

Abstract: In the medical imaging domain the market pressure and increased software content of products form the driving force for the introduction of product line architectures. This paper describes a new product line architecture that is based upon a generic platform that can only be varied through well-defined interfaces. It explains the rationale behind this approach and elaborates on the experiences so far. The paper gives a short overview of the architectural requirements for the product line and the methodology followed. The focus is on the actual platform architecture and its technical implementation.

Development/Maintenance/Reuse: Software Evolution in Product Lines

Stephen R. Schach[1] and Amir Tomer[2]
[1]Vanderbilt University, Nashville, TN, USA
[2]RAFAEL, Haifa, Israel
srs@vuse.vanderbilt.edu, tomera@rafael.co.il

Abstract: The evolution tree model is a two-dimensional model that describes how the versions of the artifacts of a software product evolve. The propagation graph is a data structure that can be used for effective control of the evolution of the artifacts of a software product. In this paper we extend the evolution tree model and propagation graph to handle the evolution of a software product line. Software product lines are characterized by large-scale reuse, especially of core assets. We show how a third dimension can be added to the evolution tree model to handle this reuse. In particular, the new model incorporates bidirectional reuse within product lines. That is, the new model can handle the transfer of an artifact from the core assets repository to a specific product (acquiring a core asset), as well as the transfer of a specific asset from a specific product to the core assets repository (mining an existing asset).

Scoping Software Product Lines — An Analysis of an Emerging Technology

Klaus Schmid
Fraunhofer Institute for Experimental Software Engineering (IESE), Sauerwiesen 6,
D-67661 Kaiserslautern, Germany
Klaus.Schmid@iese.fhg.de

Abstract: Software product line development is a rather new topic area within domain-specific software engineering that builds on previous work in domain engineering. A crucial step in developing a product line is the scoping step, which aims at determining the boundaries for the product line. This is one of the core planning activities that may determine the success or failure of the whole product line effort. In this paper, we seek to analyze the existing body of knowledge on product line scoping. As the relationship to domain engineering is very close we will also include domain scoping approaches in our analysis. Further, we will look at product line scoping related activities in other fields of study besides software engineering. As a result of this survey we provide a taxonomy of existing approaches both on a problem level, as well as on a solution level, discuss the relative advantages of the various approaches, and show some ways on using the results of this paper for enhancing existing scoping approaches and developing new approaches.

Component Based Product Line Development of Avionics Software

David C. Sharp
The Boeing Company, St. Louis, MO
david.sharp@boeing.com

Abstract: Just as hardware integrated circuits, or components, can be used to inexpensively manufacture a product line of related hardware systems, re-usable software components can be used to create software systems. This is accomplished by developing a common framework for a product line of related software systems that forms the component operating environment. A development architecture is presented based on our work using object-oriented analysis and design techniques to create reusable software components that combine with aircraft specific customizations to form an avionics software system.

The SSEP Toolset for Product Line Development
An XML Based Architecture Centric Approach

Douglas Stuart, Wonhee Sull, Steve Pruitt, Deborah Cobb, Fred Waskiewicz, and T. W. Cook
Microelectronics and Computer Technology Corporation
stuart@mcc.com, sull@sktelecom.com, pruitt@mcc.com, cobb_deborah@hotmail.com, wask@austin.rr.com, tw@acm.org

Abstract: In this paper, we describe the SSEP (Software and Systems Engineering Productivity) product line development toolset that was created to provide the tool support necessary to improve the viability of product line development. Navigation between the various artifacts that arise during architecture-based product line development is crucial to successful product line development. The SSEP toolset was developed with such navigation, provided by using XML as a representation mechanism, as a key feature. The goals of product line development are achieved through planned reuse of product line artifacts in multiple applications in the product line. Such strategic reuse depends on the application engineer's being able to develop the application within the context of the product line, which requires access to product line artifacts in their context. Such in-context access can only be provided by links capturing the dependence relationships between product line artifacts. The SSEP toolset provides such access.

Starting a Product Line Approach for an Envisioned Market
Research and Experience in an Industrial Environment

Steffen Thiel[1] and Fabio Peruzzi[2]
[1]Software Technology Center, Corporate Research and Development, Robert Bosch GmbH, Germany
[2]B4E GmbH, Germany
steffen.thiel@de.bosch.com

Abstract: At the present time, systems for the automotive market are typically developed independently from each other. The resulting systems are closed systems in the sense that they are built to perform specific operations but nothing more. Interoperability between those systems is difficult to establish. The need for a more integrated, software-based development approach is changing the way these systems are designed. Many car manufacturers and system providers are moving in this direction and it is broadly accepted that this paradigm shift in the development of automotive systems will be reflected in a very interesting new market. So far this market is an envisioned market, and specific products have not yet been developed. This paper describes a part of the process defined during the initiative undertaken by Robert Bosch GmbH and the Software Engineering Institute of Carnegie Mellon University to instantiate a product line for innovative automotive systems. The process steps described in this report specifically deal with the early phases of product line development. In particular, the experience gathered from defining and applying product line analysis and design in the domain of automotive systems is described.

A Cooperative Model for Cross-Divisional Product Development for a Software Product Line

Peter Toft, Derek Coleman, and Joni Ohta
Product Generation Consulting, Hewlett-Packard, Palo Alto, USA
peter@tofty.com, {derek_coleman, joni_ohta}@hp.com

Abstract: Organizations developing software for families of related products must meet the challenge of leveraging software development effort across the product families, while still allowing individual product teams to focus on developing their specific products. This can be a tough balancing act: too much centralization can result in slow decision making, increased time to market and conflict between product teams. Too little centralization can waste opportunities for leverage and can increase redundant and incompatible development. In this paper we explore an approach that is very similar to the traditional notion of cooperative organizations. We show how product teams are organized into a software cooperative, and examine the key roles that support this organizational model. We reason about why this model is successful. We also show how this organizational model is facilitated by an explicit software architecture, and by a specific software configuration management approach.

Strategic Product Development
A Strategic Approach to Taking Software Products to Market Successfully

Jay van Zyl[1] and A.J. Walker[2]
[1]Rubico, Johannesburg, South Africa & California, U.S.A.
[2]Software Engineering Applications Laboratory, Electrical Engineering,University of the Witwatersrand, Johannesburg, South Africa
jay@rubico.com, walker@seal.ac.za

Abstract: Product strategists are faced with difficult, but challenging tasks when it comes to product innovation, concept development and product commercialization. Product innovation must take place in order to create products and services that potential customers do not yet know they need. Different, but integrated, standards are needed in the pursuit of building capable processes for delivering these products. Strategic Product Development is an approach that uses a number of industry standards in building an organization capable of delivering commercially successful products. This paper presents an overview of the integrated approach that ties many concepts together in assisting the product development team to deliver world-class software products. Rubico Technologies is used as a case study to show how the concepts were implemented.

Remember the Basics
Key Success Factors for Launching and Institutionalizing a Software Product Line

Thomas Wappler
IBM Consulting Group, IBM Unternehmensberatung GmbH, Am Fichtenberg 1, D-71083 Herrenberg, Germany
wappler@de.ibm.com

Abstract: The concept of a software product line developed by the Software Engineering Institute is a comprehensive model for an organization building applications based on common architectures and other core assets. In a transition project for launching and institutionalizing a software product line we need to address virtually all areas of a software development organization. Such a transition project will be very difficult and there is usually very little experience in the organization in running such a project successfully. This paper discusses four factors that are essential for the transition project's success. These factors are a clearly defined goal, strong project management, strong management support and pilot projects. Although these factors seem to be very basic, experience shows that they are often neglected. In this paper we will show why they are essential for a successful transition project.

Government Product Lines

William G. Wood
Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213
wgw@sei.cmu.edu

Abstract: Government agencies often acquire large-scale software-intensive systems to be installed in many sites widely distributed across the country. These are often single-baseline systems that must be tailored to the particular environment of a site to allow for variations in both physical and functional configurations. For example, the Federal Aviation Administration (FAA) may acquire a single display system that can be installed in all en-route Air Traffic Control Centers (ARTCCS). Such systems often take a long time to develop and a significant time period to distribute to the sites; by the time the installations are completed, the system components are close to or past obsolescence since technology is continuously evolving. This inevitably leads to a future "big bang" acquisition. This paper proposes that these problems can be resolved by considering the systems to be a product line that evolves incrementally, with new technology and functionality added over the lifetime of the product line. In this way the systems will remain state-of-the-shelf and avoid a future "big bang" acquisition.

A Hierarchy of COTS Certification Criteria

Sherif Yacoub[1], Ali Mili[1], Chakri Kaveri[1], and Mark Dehlin[2]
[1]Department of CSEE, West Virginia University, Morgantown, WV
[2]The National Product Line Asset Center 1000 Technology Drive, Fairmont, WV
{yacoub, amili, kaveri}@csee.wvu.edu, dehlin@nplace.wvhtf.org

Abstract: At the same time as we recognize that generic forms of software reuse have fallen short of their expectations (in terms of gains in process productivity, product quality, and time to market), we also find that specialized forms of software reuse, such as commercial-off-the-shelf (COTS) based development and product-line engineering (PLE), have a great deal of potential in practice. To reap the benefits of practicing these two specialized forms of software reuse, COTS components can be used in product lines to streamline the development process. However, acquiring commercial components for a product line carries several risks. Testing and certification techniques are essentially required to assess the suitability of a COTS component for integration in a product-line architecture. The National Product Line Asset Center (NPLACE) is confronted with the problem of developing certification and suitability testing criteria for several COTS components in the market. In this paper, we develop a hierarchical reference model to guide the development of COTS certification criteria. We use an example of a database management system (DBMS) to illustrate the applicability of the model.

This material is being posted by Carnegie Mellon University's Software Engineering Institute (SEI) on this Web site as a community service.

The First Software Product Line Conference

Conference Tutorials

Monday, August 28

- COPA: A Component-Oriented Platform Architecting Method for Families of Software Intensive Electronic Products (Full day)
- A Feature-Oriented Method for Product Line Software Engineering (Full day)
- Designing and Evolving a Product Line Architecture (Half day, AM)
- Architecture Synthesis (Half day, PM)

Tuesday, August 29

- Architecture for Planning Software Product Platforms (Half day, AM)
- TrueScope(tm): A Full Life-cycle Approach to Develop Software Product Lines (Half day, AM)
- Product Line Architectures, Aspects, and Reuse (Half day, AM)
- Process Improvement for Software Product Lines (Half day, PM)
- Introduction to the Architecture Based Design Method (Half day, PM)
- Building Reusable Test Assets for a Product Line (Half day, PM)

Title: COPA: A Component-Oriented Platform Architecting Method for Families of Software Intensive Electronic Products
Presenter: Henk Obbink, Jürgen Müller, Pierre America, and Rob van Ommering
Time: Monday, Full day
Abstract: This tutorial will present a coherent view of a product family development approach covering subjects ranging from the business goals to integration testing. This approach is especially suited for systems that contain embedded software of considerable size and complexity. It combines modern

techniques such as component-based software development, domain analysis, platform-based reuse, etc. The approach was validated in practice in several divisions of a large industrial company and the tutorial will present two in-depth cases taken from these industrial applications.
Intended Audience: System and software architects with at least a little experience in developing product families.

Title: A Feature-Oriented Method for Product Line Software Engineering
Presenter: Kyo C. Kang
Time: Monday, Full day
Abstract: To be able to engineer application software for a product line, the application domain of the product line must first be understood in terms of application family's commonality and variability. Since the feature-oriented approach to domain analysis (FODA) was introduced in 1990 by the Software Engineering Institute, many domain engineering methods for software reuse have adopted the technique to support the domain commonality and variability analysis. FORM (Feature-Oriented Reuse Method), for instance, expanded FODA to include architecture design and object-oriented component development. This method has been applied to several industry application domains, including elevator control systems, electronic bulletin board systems, yard automation systems, and PBX, to create product line software engineering environments and software packages. This tutorial will introduce the FORM method using these industry applications as example.
Intended Audience: The tutorial will be prepared primarily for those who are interested in adopting the product line software development in their organization and those who want to develop reusable packages. Researchers who are working on software reuse will also benefit from this course greatly.

Title: Designing and Evolving a Product Line Architecture
Presenter: Jan Bosch
Time: Monday, Half day, AM
Abstract: One can identify two important developments in software architecture, i.e. software product lines and software architecture design. This tutorial addresses these topics by presenting an method for architectural design explicitly focussing on assessment of and transformation for quality attributes. In addition, the notion of software product lines and the process, technology, business and organizational issues of adopting the approach are presented. Both design and product-lines are extensively illustrated by examples and experiences from several industrial cases. The tutorial is based on a forthcoming book (tentative publication date: early June 2000) authored by the tutorial presenter and published by Addison-Wesley.
Intended Audience: The expected audience can be divided into two categories. First, software engineers and technical managers considering the introduction of software architecture design and, potentially, product-line architectures in their organization. Second, researchers interested in the experiences collected by the tutorial presenter and his research group and the reflections made based on their experiences.

Title: Architecture Synthesis
Presenter: Mehmet Aksit
Time: Monday, Half day, PM

Abstract: One of the major issues in designing architectures is to derive the fundamental abstractions of architectures from the business context and requirement specifications. This tutorial first presents a comparative analysis of the current architecture design approaches. To this aim, with in the context of Product Line architectures, OMT, Unified Process, architectural patterns, domain specific architecture design techniques are discussed. Based on the identified issues, an architecture synthesis process is presented. This process is based on mapping business context and requirement specifications into a set of concerns and synthesizing the concerns into architectural abstractions through a systematic application of solution domain knowledge.
Intended Audience: This tutorial can be particularly useful for architecture designers who want to obtain a comparative understanding of the current architecture design approaches, recognize their strong and weak points, and learn techniques to identify the fundamental abstractions of architecture using problem synthesis techniques. The participants must have an understanding of the basic object-oriented concepts and some knowledge of at least one of the popular object-oriented analysis and design methods. Further, basic knowledge on software architectures is highly recommended.
The first 10 registrants for this tutorial will receive a free copy of the book "Synthesis Based Architecture Design".

Title: Architecture for Planning Software Product Platforms
Presenter: Derek Coleman
Time: Tuesday, Half day, AM
Abstract: For companies today, creating the design and the software or firmware for each product from the ground up is no longer feasible - especially from the point of view of product quality, ease of implementation, and shorter product development schedules. Therefore, the trend is to create new product versions by intentionally reusing the architecture, design, and code from an established software platform. In this tutorial we explore the role of architecture in planning product platforms. In the planning phase, the task is to develop an architecture to support the planned product portfolio. This requires modeling the requirements and architectures of the products on a vintage chart. Because there is always uncertainty about the future, the planning must be lightweight enough to be done quickly yet accurate enough to be worth doing. Planning product line architectures needs modeling techniques that have just enough precision and formality, i.e. lightweight and focussed on key needs.
Intended Audience: The tutorial is aimed at architects, software engineers, project managers and process engineers interested in how to develop architected software platforms. Preferably participants should have some familiarity with software modeling notations and techniques, e.g. UML, use cases etc.

Title: TrueScope(tm): A Full Life-cycle Approach to Develop Software Product Lines
Presenter: Jean-Marc DeBaud
Time: Tuesday, Half day, AM
Abstract: The goal of this tutorial is to provide the participants with the basics of an overall, well-integrated approach to develop software product families they can use in practice. The key of our process resides in its ability to define the software family's potential economies of scope at the requirements level and, as strictly as possible, propagate these economies to the reference architecture (the platform). The platform is the most critical asset of a product family. Its scope both determines its ability to handle the variation among the different product family members but also really how much

return on the overall investment can be made by switching the organization to a product family engineering mode of operation. Hence, gaining a quantitative understanding of the tradeoffs involved when including or excluding products or features within the family and/or platform is critical to the success of any product family engineering endeavor. For more information, please see www.truescope-tech.com.
Intended Audience: This tutorial is open to everyone.

Title: Product Line Architectures, Aspects, and Reuse
Presenter: Don Batory
Time: Tuesday, Half day, AM
Abstract: This tutorial is aimed at researchers and practitioners who are faced with the problem of building Product-Line Architectures (i.e., families of applications) economically, and/or who are interested in assembling customized and evolvable applications automatically from component reuse libraries. We present an increasingly popular model of architecturally-evolveable software (i.e., software that evolves through the addition and removal of components) that brings together significant research areas: aspect-oriented programming, Perry's lite semantics (for testing architectural consistency), parameterized programming, generative programming, and Baxter's design maintenance. Among the case studies presented in the tutorial are extensible Java compilers and extensible simulators for Army fire support.
Intended Audience: The tutorial assumes rudimentary knowledge of OO concepts, but no experience with the tutorial's subject.

Title: Process Improvement for Software Product Lines
Presenter: Grady Campbell
Time: Tuesday, Half day, PM
Abstract: The goal of this tutorial is to present a systematic process improvement method that organizations can follow to adopt a product line approach to software development. Starting with an organization's business objectives and technical expertise, this method helps adopters evaluate the viability of a product line approach, improve process and reuse maturity to an acceptable level, target a beneficial and cost-effective level of reuse capability, and implement a strategy for instituting corresponding product line practices.
Intended Audience: This tutorial is targeted to experienced engineers and managers whose organizations repeatedly build similar products and want to achieve substantial improvements in software productivity and product quality. Tutorial attendees should be familiar with the principles and use of a conventional process improvement method such as the SEI Capability Maturity Model approach.

Title: Introduction to the Architecture Based Design Method
Presenter: Felix Bachmann and Len Bass
Time: Tuesday, Half day, PM
Abstract: This tutorial will introduce the Architecture Based Design (ABD) method. This is a method for designing the conceptual architecture for a product line. The conceptual architecture includes the

decomposition of function for the final system, identification of possible threads of parallelism and identification of possible physical network configurations and the allocation of the functional decomposition to processors.
The ABD method includes explicit steps that enable the designed architectures to serve as the basis for product lines of software. These steps include explicit consideration of commonalities and variabilities and explicit consideration and verification of quality and business requirements. The ABD method is based on the identification of architectural drivers (those requirements that "shape the architecture") and this allows the architecture design to be carried on in parallel to the requirements process.
The ABD explicitly supports the design of product lines and it is one of the few design methods that makes explicit the dependence of architecture on quality and business requirements.
Intended Audience: System architects and designers who want to understand how to design architectures, especially product line architectures) that achieve the desired system qualities.

Title: Building Reusable Test Assets for a Product Line
Presenter: John D. McGregor
Time: Tuesday, Half day, PM
Abstract: The testing activities on a project consume valuable resources that could better be used to increase functionality. This tutorial presents techniques for taking advantage of the personnel organization and software architectures to reduce the effort required for adequate testing. The techniques organize unit-level testing assets in a manner that directly reflects the architecture of the product software. The techniques also associate the requirements in the form of use cases with the system test cases.
Intended Audience: Managers, system testers, quality assurance personnel, and development personnel. This tutorial will assume that the participant has a developer's working knowledge of testing terminology and concepts.

The First Software Product Line Conference

Conference Workshops

Important: Workshop Attendance Criteria

Monday, August 28

- Combining Architecture, Asset Management, and Culture to Successfully Develop Product Lines
- Product Line Architecture
- Architecture Reconstruction and Product Lines
- Measurement Issues for Software Product Line Engineering

Tuesday, August 29

- Generative Techniques for Product Lines
- Product Lines for Command-and-Control Ground Systems
- Embedded System Product Lines

Combining Architecture, Asset Management, and Culture to Successfully Develop Product Lines

A product line development solution (PLDS) consists of three aspects: an architecture, a software asset management strategy and a social/organizational structure. To be successful at moving from a product focused development paradigm to a product line focused paradigm it is important to design all three elements of the PLDS to match the business situation (business needs, technology, and culture). Each of the three elements has valid solutions that differ greatly. Architectures can vary from a highly optimized code set to a highly decoupled code set. Asset management can vary from a central group control to an open source model. Organization strategies can vary from reuse group focused to project focused.

We believe that there has been much exploration and progress in each of the three elements of the PLDS, but there has been little focus or discussion on how the three elements need to be combined to provide a successful approach for various business situations. This workshop will focus on the full solution instead of any one element.

Prospective workshop participants are required to submit a 2 - 3 page position paper describing either:

- Their experiences with implementing an architecture, or an asset management strategy, or a social/cultural solution.
- Their current business situation and reasons for exploring PLDS.

We will select the participants based on the quality and relevance of their position papers. We will limit the number of participants to 15, and the number of presentations to 5

Workshop Organizers:

Phil McCoog (primary contact)
Hewlett Packard, Vancouver Division
+1 (360) 212-8296
phil_mccoog@hp.com

Joe Bauman
Hewlett Packard, Vancouver Division
+1 (360) 212-2749
joe_bauman@hp.com

Product Line Architecture

A product-line architecture must adequately capture and support the description of commonalities and variabilities in the solutions. However this is not enough; variation points in the architecture, for example, are not sufficient to support the decisions that lead to a particular choice of variant. A decision model, the decisions and the relationships among them, defines a set of decisions that are adequate to distinguish among members of application family and to guide adaptation of adaptable architecture and its components. The objective of this workshop will be to investigate potential technologies necessary to support the derivation of architecture variants. Two main focus areas will be addressed:

- Technologies to support variability: Variation points and variability mechanisms (e.g., parameterization, generics, etc.) and criteria for mechanism selection.
- Technologies to support derivation of variants: the decision model (decisions and relationships among them for variant derivation).

Workshop Organizers:

Jean Jourdan (primary contact)
Thomson-CSF/LCAT
Domaine de Corbeville
91404 ORSAY France
Phone: +33 (1) 69 33 09 18
FAX : +33 (1) 69 33 08 65
jourdan@lcr.thomson-csf.com

Michel Coriat
Thomson-CSF/LCAT
Domaine de Corbeville
91404 ORSAY France
Phone: +33 (1) 69 33 07 76
FAX : +33 (1) 69 33 08 65
coriat@lcr.thomson-csf.com

Dr Cristina Gacek
Fraunhofer Institute for Experimental Software Engineering (IESE)
Sauerwiesen 6
D-67661 Kaiserslautern Germany
Phone: +49 (6301) 707-224
FAX: +49 (6301) 707-200
Cristina.Gacek@iese.fhg.de

Architecture Reconstruction and Product Lines

This workshop will focus on architecture reconstruction in product line contexts. In most cases product lines evolve out of existing products. To manage the evolution in a disciplined way organizations have to 'mine' existing assets for architecture, commonalities, and varieties. Reconstruction focuses in most cases on an abstraction of at least a few components and interactions to present an understandable structure. But this focus is not sufficient in the context of product lines. To detect reusable components and varieties for a potential product line migration there are further challenges: (1) The reconstruction process should be applied to at least one further product and (2) the analyst has to find a useful abstraction level to detect and describe the commonalities and varieties. Summarized the objectives of this workshop are:

- Experiences and techniques for finding commonalities and varieties in legacy software.
- Tool environments and techniques for finding commonalities and varieties.
- Benefits and costs of architecture reconstruction.
- Open technical issues.

Participants are invited to submit a position paper with a contribution on one or more of the objectives. The workshop will consist of presentations of these papers in the morning session, followed by an afternoon session of discussion and collaboration.

Workshop Organizers:

Christoph Stoermer (primary contact)
Robert Bosch Corp.
Research and Technology Center
+1 412-268 3949
cstoerme@sei.cmu.edu

William O'Brien
Software Engineering Institute
+1 412-268 7727
lob@sei.cmu.edu

Measurement Issues for Software Product Line Engineering

Effective management and improvement of software development requires the support of a good measurement system. While measurement for software engineering is not new, it appears that

measurement in support of a product line approach to software engineering does pose some unique challenges. This workshop will identify and discuss:

- Management roles associated with a product line approach to software engineering
- Management issues and responsibilities associated with the management roles
- Measures to address the issues and needs
- Measurement practices to provide the measurement data

Much work and guidance exists on software measurement. More recently, measurement in support of software reuse has appeared in the literature. The concept of a product line approach to software development is still evolving, however, and little work has been done to date to describe the measures and measurement practices that are needed to efficiently and effectively manage and improve a software product line operation.

For measurement to be useful and effective, it needs to support action and decision-making. In the operation of a software product line, there are at least distinct managerial roles to be addressed: the overall product line operation, asset development and management, and application development and maintenance. We will investigate the scope and responsibilities of the managerial roles and the characteristics of a measurement system that supports these roles.

Prospective workshop participants should submit a 2 - 3 page position paper describing their experiences with software product line management and/or measurement. We will select the participants based on the quality and relevance of their position papers. We will limit the number of participants to 15, and the number of presentations to 5.

Workshop Organizers:

Dave Zubrow (primary contact)
Software Engineering Institute
+1 412-268-5243
dz@sei.cmu.edu

Grady Campbell
gradycampbell@acm.org

Wolf Goethert
Software Engineering Institute
+1 412-268-3889
wbg@sei.cmu.edu

Generative Techniques for Product Lines

The workshop on Generative Techniques for Product Lines aims to bring together practitioners, researchers, academics, and students to discuss broadly the state-of-the-art of generators and their role in developing a product line. Topics of interest, include:

- styles of generators (application generators, template-based generators,
- transformational systems, intentional languages, aspects, subjects, etc), particularly their uses and limitations;
- generation of code artifacts, such as application logic, UIs, database schemas, and middleware

integration;
- generation of non-code artifacts such as test cases, documentation, tutorials, and help systems;
- capturing configuration knowledge, for example, in DSLs, and extensible languages; and
- testing generic and generative models.

The goal is to share experience, consolidate successful techniques, and identify open issues for future work in product lines.

Potential participants are asked to submit a two-page position paper detailing their experience with generative techniques, their perspective on one or more of the above topics, and their planned contribution to the workshop. Based on the position papers, the organizers will invite a cross-section to participate.

Workshop Organizers:

Greg Butler (primary contact)
Concordia University, Montreal
gregb@cs.concordia.ca

Krzysztof Czarnecki
Software Technology Lab
DaimlerChrysler Research, Ulm
czarnecki@acm.org

Ulrich Eisenecker
University of Applied Sciences
Kaiserslautern
Ulrich.Eisenecker@t-online.de

Product Lines for Command-and-Control Ground Systems

This workshop will address issues relating to implementation of product lines for C2 ground systems. Although the focus is not exclusively on C2 ground systems for spacecraft, discussions will begin with that domain, since the organizers of this workshop have also been organizers of the Ground System Architectures Workshops (GSAW), which have drawn their attendees primarily from the spacecraft C2 domain. Issues relating to command-and-control ground systems in other domains will also be addressed in this workshop, especially with regard to how they are similar to and different from issues in the spacecraft ground systems domain.

This workshop will begin with a focus on the following issues in the context of command-and-control ground systems:

- What experiences have participants had with the definition, development, and evaluation of product line architectures?
- What evaluation techniques will help to identify/define the architectural variability needed for a product line?
- What architectural features should be captured using architecture description languages to facilitate evaluation of a product line architecture?
- How much of a product line architecture should be common across all the systems, and how much should be system-specific? What techniques can be used to help identify an appropriate common "core" architecture?

- What is the impact of the stakeholder set on the definition of a common "core" product line architecture?
- Where do COTS products fit in the development of a product line?
- What are the most critical issues that remain open?

Prospective workshop participants should submit a 1 - 2 page position paper describing their experiences with software product lines for command-and-control ground systems. The participants will be selected based on the quality and relevance of their position papers.

Workshop Organizer:

Judy Kerner
The Aerospace Corporation
MS M1/113 POB 92957
Los Angeles, CA 90009
+1 (310) 336-6555
Judy.Kerner@aero.org

Embedded System Product Lines

A product line is a group of products sharing a common, managed set of features that satisfy specific needs of a selected market. Once a product line has been developed, managing the product line to balance core and application specific content and changes can be challenging. As new product requests are encountered, requirements need to be compared with current core content to determine if the product should be added to the product line and which parts should be added to the core and which parts should be application specific. The organizational structure must be in place to facilitate these decisions. Application teams should not have too many resources or too much freedom to make application specific changes or else core product line assets won't be used effectively. As the product gets closer to production, there will be a need to freeze the content for an individual application. This means the same change may need to be made for a specific application and later in the core so that when the application takes a future core release the content will be included. A strategy needs to be in place for coordinating core and application releases throughout the product line life-cycle.

The workshop is focused on managing the product line once it has been developed. The problems to be addressed include:

- What are the technical architectures for constrained environments (e.g. limitations on cost, throughput, and memory)?
- What are the relationships between core and applications? Is core content completely reused or is there some smaller percentage of reuse? How is reuse monitored and measured? How is core and application content maintained through the product life cycle?
- What are the cultural roadblocks to be encountered and how are they mitigated?
- How are multiple products with different delivery dates coordinated with core releases?

Prospective workshop participants are required to submit a 2 - 3 page position paper describing their experiences with software product line analysis in this field. We will select the participants based on the quality and relevance of their position papers. We will limit the number of participants to 15, and the number of presentations to 5.

Workshop Organizers:

Kathy Rose (Primary Contact)
+1 (248) 684-3726
krose01@powertrain.mpg.gm.com

Jim Kolhoff
+1 (810) 220-2595
jkolho01@powertrain.mpg.gm.com

Jim Kurnik
+1 (810) 220-2458
jkurni01@powertrain.mpg.gm.com

Workshop Attendance

Workshops are by invitation only. Each workshop may have its own invitation criteria. The most likely criteria for attendance will be the submission and acceptance of a workshop paper. Workshop papers tend to be much less formal and comprehensive than main-conference papers.

If you are interested in attending a specific workshop, please follow the invitation criteria provided with the workshop and contact the workshop organizer(s).

The First Software Product Line Conference

Keynote Speaker

Colin Tully
European Software Process Improvement Foundation

Colin Tully is Technical Director of the ESPI Foundation. He graduated in Economics from Cambridge University in 1960, and in 1961 he was one of a two-man team who wrote the world's first multi-programming operating system (for the LEO III series of computers). For nearly thirty years, including periods at the London School of Economics and the University of York, he gained a wide variety of training, education, research, consultancy and management experience in both information systems and software engineering, latterly specialising in databases and software engineering environments. In 1989 he established Colin Tully Associates as an independent consultancy in software process improvement, working extensively in the European ESPRIT and ESSI programmes (Commission reviewer and evaluator, and partner and subcontractor in many projects). He is a European Engineer, UK Chartered Engineer, Fellow of the British Computer Society, editor and co-author of *Improving Software Practice* (Wiley, 1998), co-author of *Réutilisation Logicielle* (Eyrolles, 1999), and an editor-in-chief of the journal *Software Process: Improvement and Practice*.

The First Software Product Line Conference

Conference Panels

"*Architecture Design Methods for Software Product Lines*"

Moderator:
Robert L. Nord, Siemens Corporate Research

Panelists:
Len Bass, Software Engineering Institute
Jan Bosch, University of Karlskrona/Ronneby
Christine Hofmeister, Siemens Corporate Research
Kyo C. Kang, Pohang University of Science and Technology
Rob van Ommering, Philips Research Laboratories

A software product line is a group of software products sharing a common, managed set of features that satisfy the needs of a specific market. Business context, technology, and quality attribute requirements all exert a strong influence on the shape of the architecture for a product line. It has been postulated that what makes architectures for a product line different from those for any large system is the emphasis on the support for variation. A product line architecture must capture the description of commonalities and variabilities in the software products at a given point in time.

The issue facing this panel is how well do existing design methods support the development of architectures for product lines and what remains to be done. Panelists will discuss the difference between product line architectures and architectures for single systems and how their design method exploits the difference.

"*Institutionalizing Software Product Lines*"

Organizers : Jean Jourdan, Jan Bosch

Panelist:
David Weiss, Luccent Bell Labs
Ben Pronk, Philips Research Laboratories
Juha Kuusela, Nokia
Len Bass, Software Engineering Institute

Software product lines have received increasing amounts of attention especially from industry. During recent years several software development organizations have launched software product line initiatives, in order to study, adopt and assess the opportunities and benefits offered by a product line approach in their business area. The objective of this panel session is to surface and discuss the various ways followed by these organizations to institutionalize software product lines. This panel presents the opportunity to, among others, debate on:

- How do industrial organizations adopt a product line approach ? Is it a bottom-up or a top-down initiative?
- What organizational model is applied? Do they have launched initiatives in some business divisions.
- What is the maturity of these initiatives, e.g. experiment, pilot product line or operational product line?
- What kind of process has been followed to evolve product line experiments to operational product lines?
- What migration phases are followed?
- What decision process was used to decide on the adoption of a product line approach?
- What is still lacking in terms of process, technology, tools, assessment model, adoption process, etc.?

Combining Architecture, Asset Management, and Culture to Successfully Develop Product Lines

8/30/00

Click here to start

Table of Contents

Product Line Architecture

8/30/00

Click here to start

Table of Contents

Architecture Reconstruction and Product Lines

8/30/00

Click here to start

Table of Contents

Measurement Issues for Software Product Line Engineering

8/30/00

Click here to start

Table of Contents

SPLC1 Workshop on Product Lines for Command and Control Ground Systems

8/30/00

Click here to start

Table of Contents

Embedded System Product Lines

8/30/00

Click here to start

Table of Contents

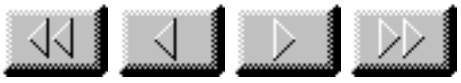
Combining Architecture, Asset Management, and Culture to Successfully Develop Product Lines

The First Software Product Line Conference (SPLC1)

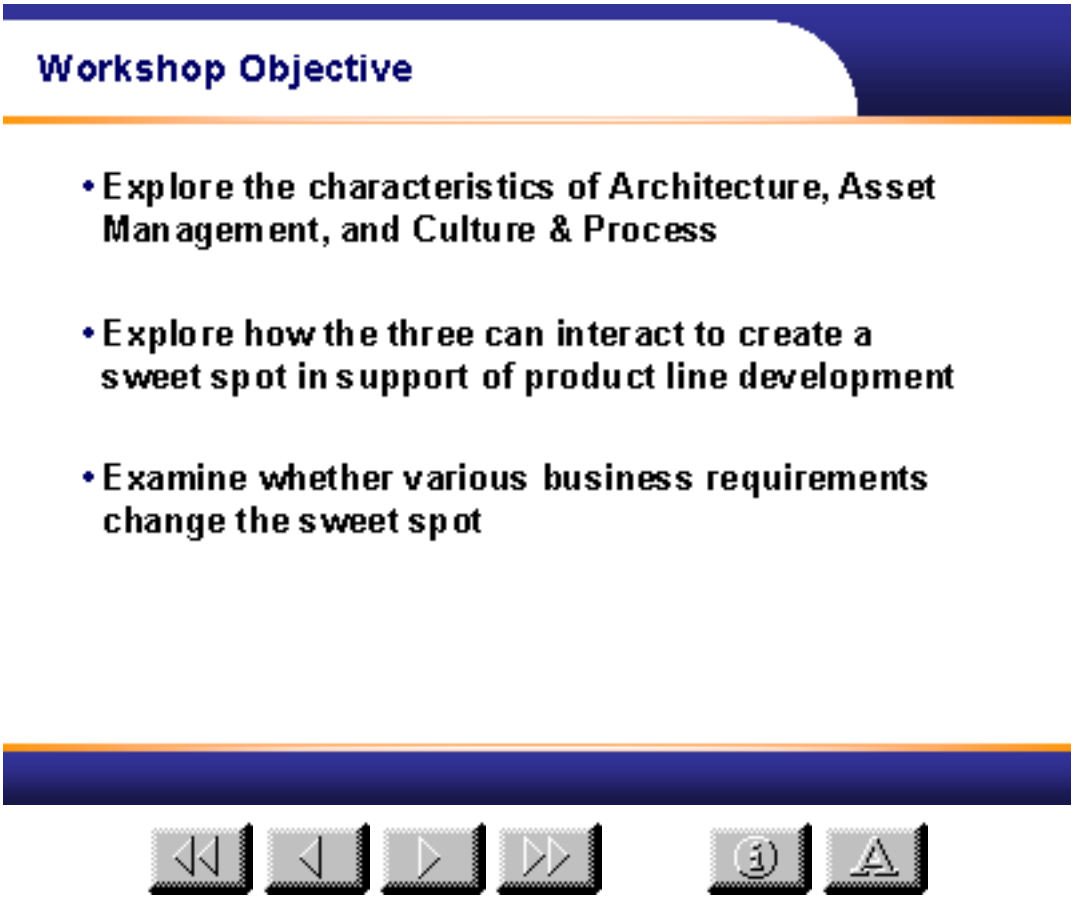
Denver, Colorado
August 28, 2000

Joe Bauman (joe_bauman@hp.com)
Firmware Architect
Hewlett Packard, Vancouver Division

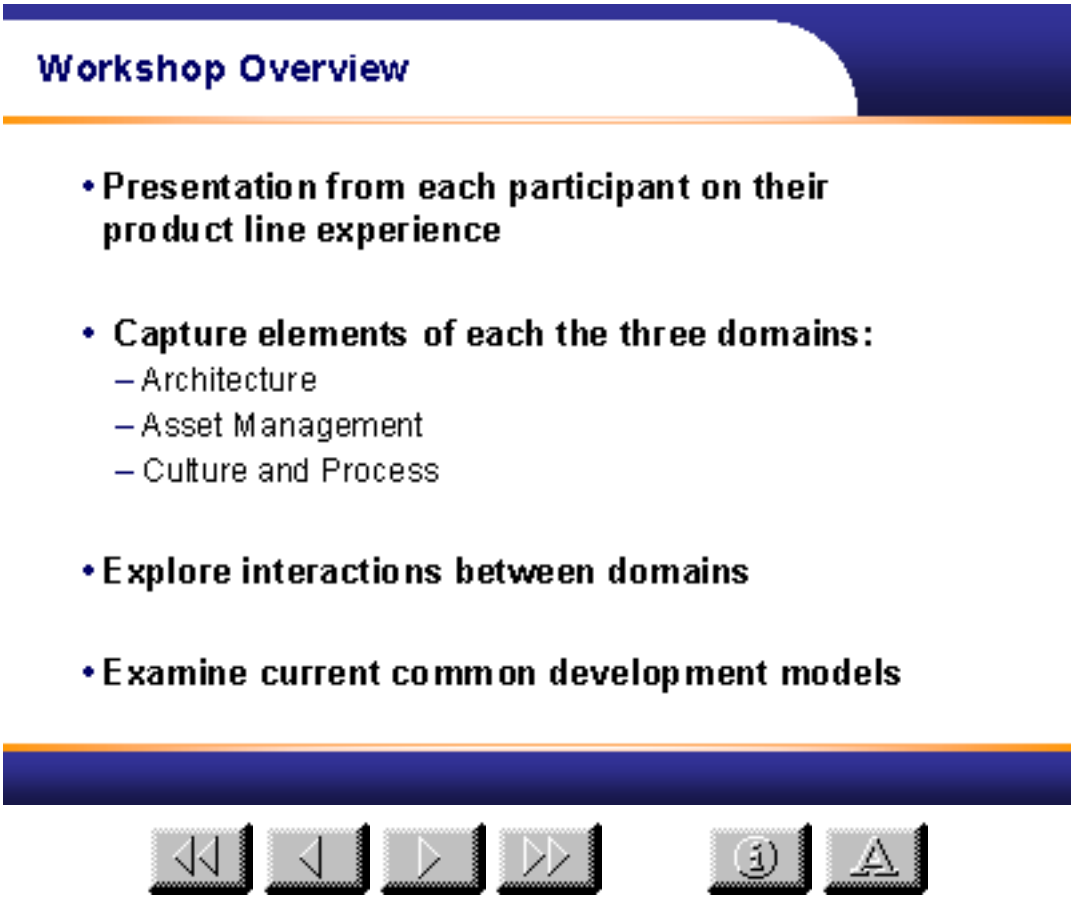
Phil McCoog (phil_mccoog@hp.com)
Firmware Asset Lead
Hewlett Packard, Vancouver Division



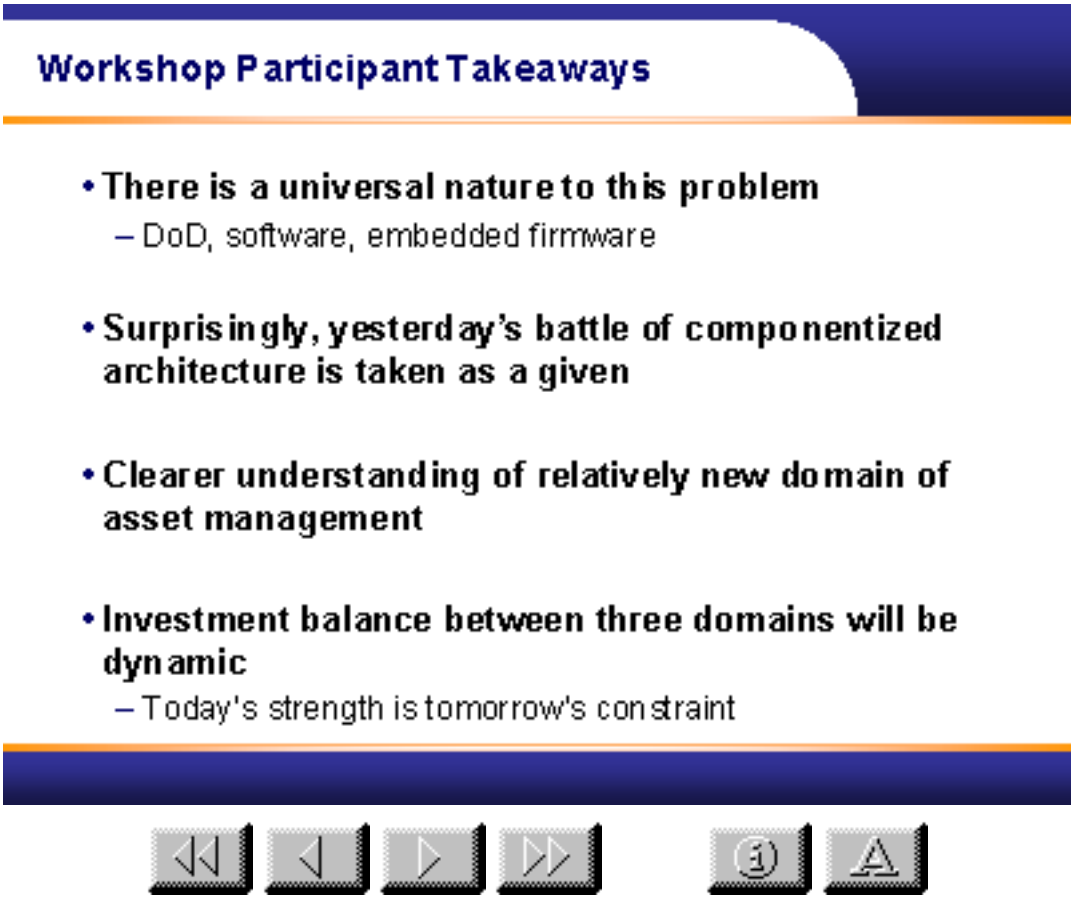
Slide 1 of 5


Workshop Objective
• Explore the characteristics of Architecture, Asset Management, and Culture & Process
• Explore how the three can interact to create a sweet spot in support of product line development
• Examine whether various business requirements change the sweet spot

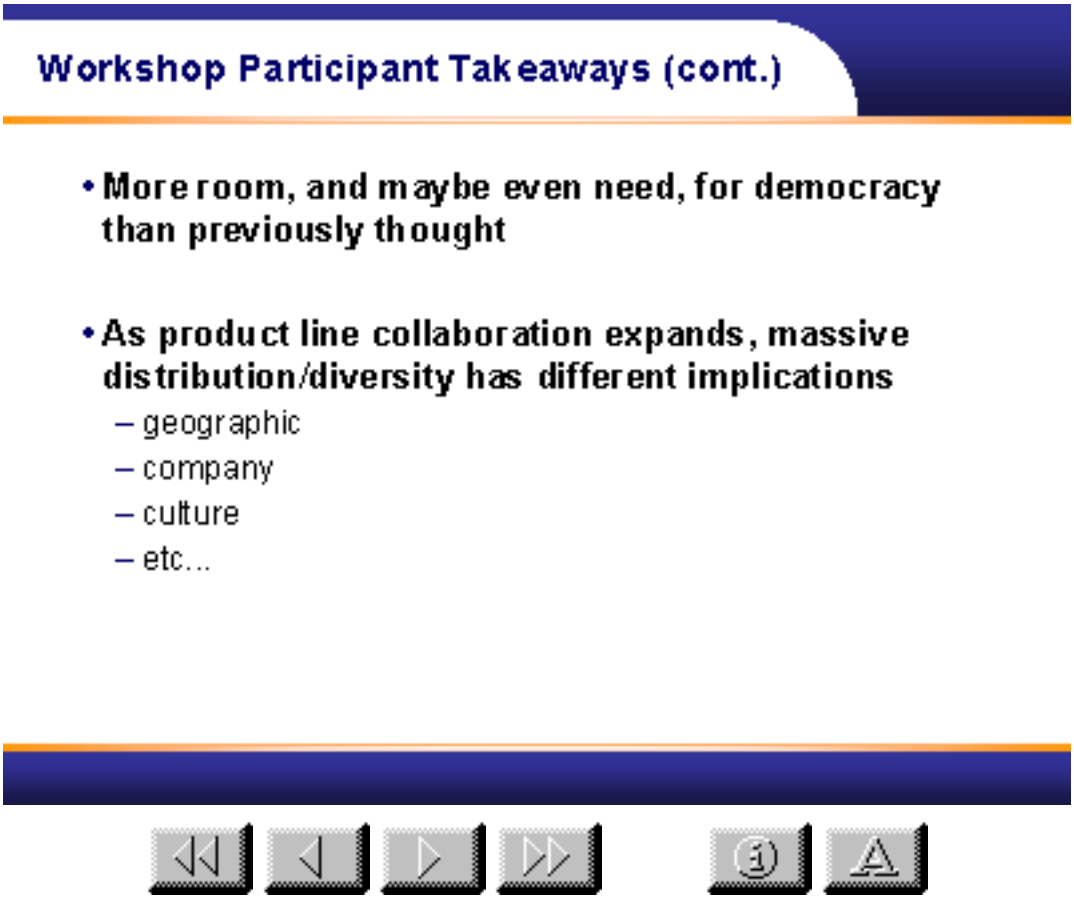
Slide 2 of 5


Workshop Overview
• Presentation from each participant on their product line experience
• Capture elements of each the three domains:
– Architecture
– Asset Management
– Culture and Process
• Explore interactions between domains
• Examine current common development models

Slide 3 of 5


Workshop Participant Takeaways
• There is a universal nature to this problem
– DoD, software, embedded firmware
• Surprisingly, yesterday's battle of componentized architecture is taken as a given
• Clearer understanding of relatively new domain of asset management
• Investment balance between three domains will be dynamic
– Today's strength is tomorrow's constraint

Slide 4 of 5


Workshop Participant Takeaways (cont.)
• More room, and maybe even need, for democracy than previously thought
• As product line collaboration expands, massive distribution/diversity has different implications
– geographic
– company
– culture
– etc...

Slide 5 of 5

First Software Product-Line Conference

Summary of workshop #2

"Product Line Architecture"

Wednesday, August 30, 2000
Denver, Colorado

Organizers:

J. Jourdan	Thomson-CSF
C. Gacek	FhG IESE
M. Coriat	Thomson-CSF

Main focus of PLA workshop

Workshop objective:

- to investigate potential technologies supporting the derivation of architecture variants.

Two main focus:

- Technologies to support variability: Variation points and variability mechanisms (e.g., parameterization, generics, etc.) and criteria for mechanism selection,
- Technologies to support derivation of variants: the decision model (decisions and relationships among them for variant derivation).

Product Line Architecture = Software Architecture + Variability, Derivation

Technology	Domain
Variation points, variability mechanisms, ...	Contain all Variant solutions
Decision model representation, Algorithms to use it, ..	Decision model

Information about the reviews

Some numbers about the submissions:

- Number of submissions: 18
- Number of reviews per paper: 2 or 3

- European papers: 6
- North American papers :12
- Industry papers: 12
- Academic papers : 6

Accepted papers: 11
6 from EU over 6 , 5 from NA over 12

- Is it because the program committee was mostly European or is there a more fundamental question behind ?

- Is the American approach of PLA slightly different from the European approach of PLA ?

Slide 3 of 13

Presentations (1/2)

10 presentations, about 25 attendees

Concepts supporting product line architecture

- Architectural evolution of product families
  A. Maccari
- Using a decision model to support product line architecture modeling, evaluation and instantiation
  O. Flege
- Managing variability in the LCATS PLIT/DAISY model
  W. El Kaim

Component and computing infrastructure for PLA

- Diversity interfaces, variation and variation management
  F. van der Linden
- Generic software component configuration via partial evaluation
  A. F. Le Meur, C. Consel

Presentations (2/2)

Technology supporting PLA development

- Containing and facilitating change via object oriented tailoring techniques
  D. Sharp
- Pattern languages for product line customization
  G. Butler
- Documenting and controlling product lines using the UML
  D. Muthig

Practical experiences on PLA

- An OO application frameworks perspective on commonality and variability in PLA
  H. Hoover, T. Olekshy, G. Froehlich, P. Sorenson
- Addressing variability in a guidance, navigation, and control flight software PL
  D. McComas, S. Leake, M. Stark, M. Morisio, G. Travassos, M. White

Working groups

We proposed to set up 3 working groups

Suggested working group #1 : Variability
Suggested working group #2 : Evolution and traceability
Suggested working group #3 : Derivation

Anecdote

Nobody was willing to joint the "derivation" working group (one of the two main focus of the workshop)

WHY ???

I got 2 explanations:

- There is a wording problem: do not use derivation but rather generation or instantiation
- It seems to be too early: derivation arrives later in the process

Breakout groups

Working group #1: Variability
Working group #2: Evolution and traceability
Working group #3: Other issues

- Various brainstorming and idea production on:
  - PLA and Architecture of a single system
  - Variability

Active discussion on

PLA versus Architecture of a single system

Is there really a difference ?

- PLA architecture contains
  - the commonalities of several products (it is a part of the architecture of the single systems)
  - plus, in an **explicit and documented way**, the variabilities of all the products
- Two different trends for the concept of PLA
  - Application oriented PLA yields to a final system
  - Infrastructure oriented PLA aims at facilitating component integration

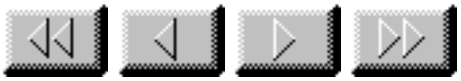
Slide 8 of 13

Variability versus Flexibility

- Different kinds of variability:
  - required variability: it comes from requirements
  - induced variability: dependency between variability does exist
  - acquired variability: yet acquired variability before building the product line
- **Variability should not be confused with Flexibility**:
  - **variability** is the difference between several products in the same family,
  - **flexibility** is related to the capacity of an architecture to extend and satisfy new needs and requirements.

Customisable products and product families are two different things (Some disagreement on this)

This point has to be made clearer to the whole product line community

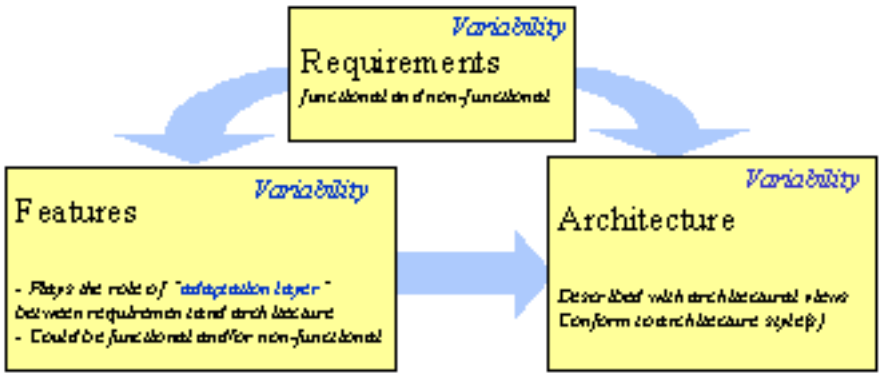


Slide 9 of 13

Variability in architecture

- Current Status:
  - Architecture is described using a set of architectural views (no consensus on set of views).
  - Variability is present in all architectural views.
  - In each view, assets can vary:
    - existence, dependency, relationships, lifecycle, static/dynamic, induced/explicit.
  - Variation in assets is represented by several notations (Variation Point, Feature Model, Pattern, Parameters).
- Next Step:
  - How to rationalize the selection of the right variability mechanisms to a specific kind of variability in a view ?

Mapping variability between assets

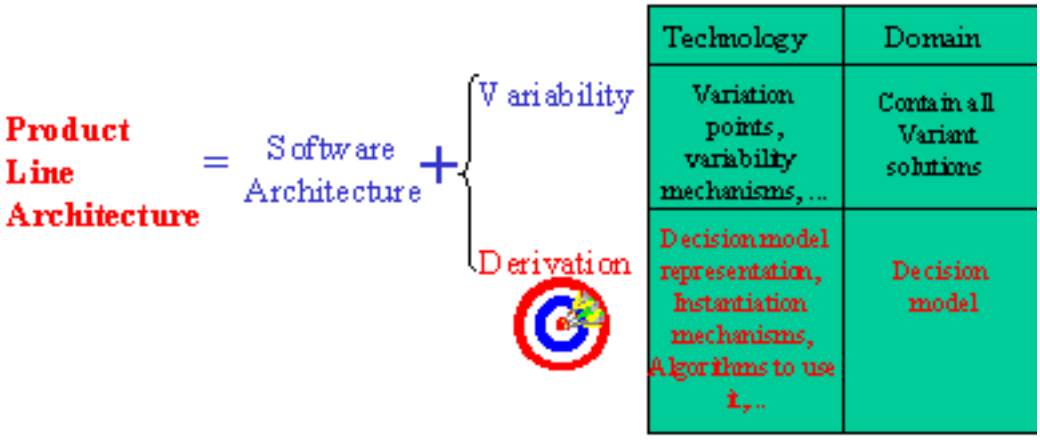


- How to map or show impact of variability between assets ?
- How to manage the impact of the variability in the number of traceability links ?

Other statements

- Scope vs Boundaries:
  - Scope: The scope defines the business and technological limited area taken into account for building a product line
  - Boundaries: Boundaries define the limit where the system wide properties hold. Going beyond such boundaries will break the architecture
  - Guideline: It is good to build product families in a way that the scope can be expanded greatly and still be within the boundaries
- Context-free design decisions: design decisions that you can combine freely, so that they hold for all components and all systems in the family independent of the context.
- Design principle: You can use principles such as "functional decomposition always generates a poor architecture" e.g. to assess a product family architecture.

Next workshop



WS – Architecture Reconstruction and Product Lines (PL)

Workshop Topics

- Experiences and techniques for identifying and evaluating commonalities and variabilities
- Benefits and costs
- Open technical issues
- Tool environments

Slide 1 of 7

WS – Architecture Reconstruction and PL

Relationship between Architecture Reconstruction and PL

- Product Lines evolve out of existing products
- Identification of commonalities and variabilities across existing products
- Evaluation of commonalities and variabilities for
  a) Technical foundation for PL decisions
  b) Risks, Sensitivity points
  c) Reference Architectures
  d) Identification of reuse potential

Slide 2 of 7

WS – Architecture Reconstruction and PL

Discussions:

- Specific foci for Reconstruction in a Product Line context. Insights:
  - Hard-wired customer features
  - Hard-wired dependencies to OS/HW/UI
  - Terminology across products
  - Software component granularity
  - Identifying a comparable set of components across products

Slide 3 of 7

WS – Architecture Reconstruction and PL

- What characteristics do PL architectures have?

  Examples of what styles/structures support specific PL architectural aspects:
  - Isolation of HW (processor, peripherals, …)
  - Communication infrastructure
  - Layering
  - Flexible configuration/calibration
  - Operating system flexibility
  - Virtual devices (indirection)

Slide 4 of 7

WS – Architecture Reconstruction and PL

- **Experiences and techniques for identifying and evaluating commonalities and variabilities**
  - Presentation of manual architecture reconstruction in a defibrillator PL context
  - Presentation of MAP (Mining Architectures for PL evaluations) in an automotive PL context

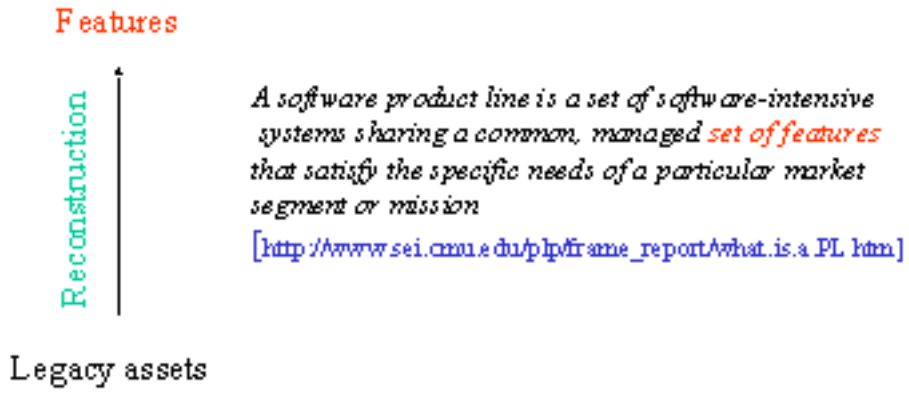
Slide 5 of 7

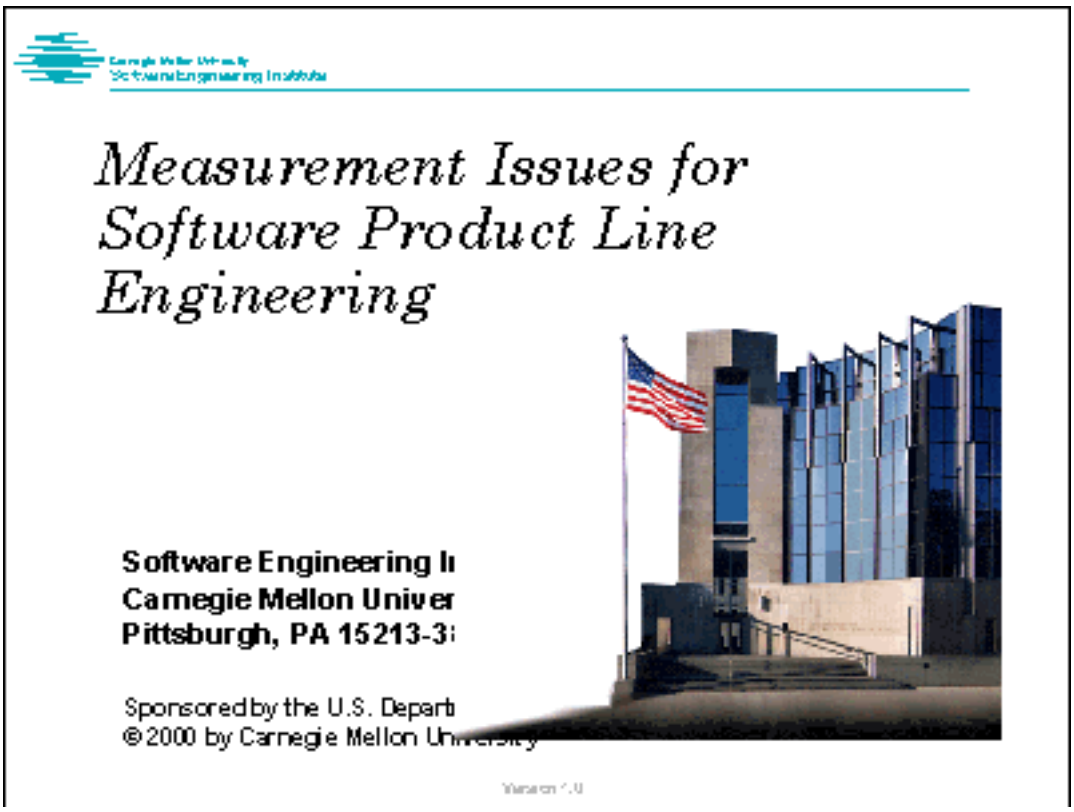
WS – Architecture Reconstruction and PL

- **Tool environments**
  - Overview of the Dali workbench, which supports architecture reconstruction
  - Experiences with the RTM tool in an acquisition environment, which supports requirement traceability

WS – Architecture Reconstruction and PL

- **Open technical issues**
  - Reconstruction to identify feature sets




Measurement Issues for Software Product Line Engineering
Software Engineering I
Carnegie Mellon Univer
Pittsburgh, PA 15213-3
Sponsored by the U.S. Depart
©2000 by Carnegie Mellon

Slide 1 of 7

Participants

- Khaled El-Emam - National Research Council, Canada
- Stefan Ferber - Bosch
- Wolf Goethert – SEI
- Cheryl Jones – Practical Software Measurement
- Klaus Schmid - Fraunhofer Institute
- Colin Tully – ESPI
- Dave Zubrow – SEI

Slide 2 of 7

Agenda

8:30 Introductions and Review of Workshop Agenda

8:45 Presentations

10:00 Break

Noon Lunch

1:30 Discuss Topics

3:00 Break

4:45 Generate Results Briefing

- Results
- Next steps

5:30 Dismiss

Slide 3 of 7

Presentations

- Product Line Indicators and Measures – Dave Zubrow
- Evaluating Component Quality – Khaled El Emam
- Scoping Software Product Lines: Using a Quantitative Approach – Klaus Schmid
- Measurement Issues for Software Product Line Engineering – Colin Tully
- Experiment at Robert Bosch GmbH: Case Study of a Product Line for Driver Assistance Systems – Stefan Ferber

Slide 4 of 7

Discussion Topics

How Do You Know You Should Adopt a Product Line Approach?

Costs and Benefits of a Software Product Line

Unique Aspects and Challenges for Measurement associated with a Software Product Line

Factors Affecting Implementation of a Software Product Line

Summary - 1

- Many of the measures for product lines are the same as they are for traditional project and organizational management
- Potential benefits address a lot of software engineering problems
- Regarding the decision to adopt a product line approach, many benefits don't have associated measures
- Both business and technical issues must be addressed - need to distinguish between business and technical measures
- Knowing current situation is important as input to transition cost calculations

Slide 6 of 7

Summary - 2

- Total product cost needs allocation from asset development
- Level of granularity of measures and more views (groupings)
- Aggregate across multiple products may be misleading, ensuring data consistency

Slide 7 of 7

SPLC1 Workshop on Product Lines for Command and Control Ground Systems

30 Aug 2000

Slide 1 of 10

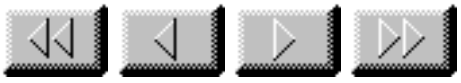

SPLC1: C2 Ground Systems Workshop
Participants (1)
Organizers
Judy Kerner
The Aerospace Corp.
Mark Walker
The Aerospace Corp.
Outbriefer
Mike Hogan
The Aerospace Corp.
Presenters
Don Batory
University of Texas, Austin
Paul Davis
TRW
Susan Kurtik
Jet Propulsion Lab.
Christopher Mushenski
US Army TACOM
John Ohlinger
Natl Reconnaissance Off.
Bob Schwanke
Siemens Corp. Res.
Kenneth D. Shere
The Aerospace Corp.
2

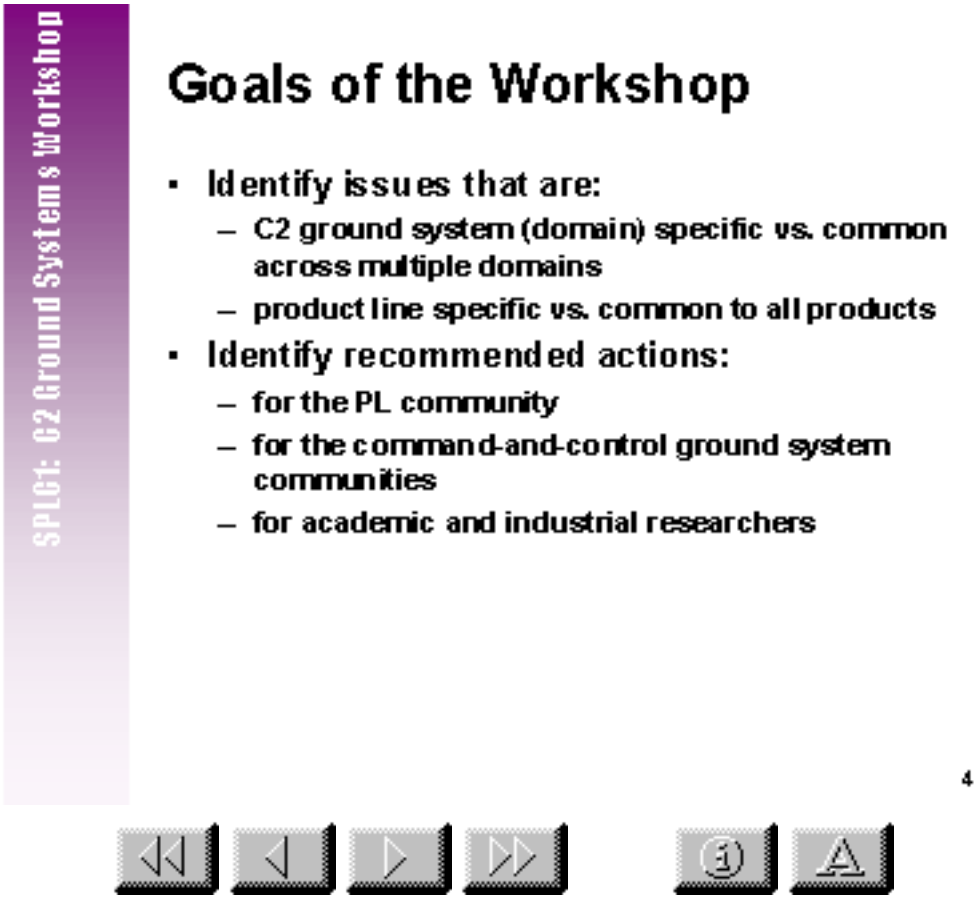
Slide 2 of 10

Participants (2)

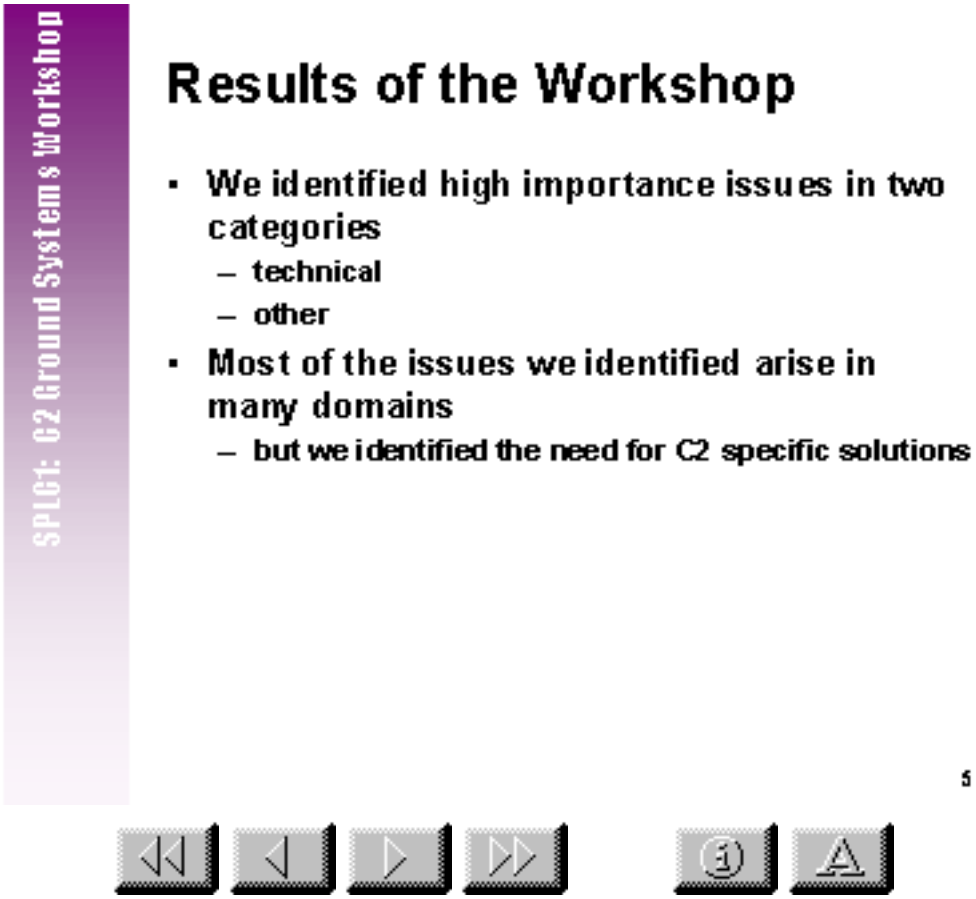
Lucy Carpenter
Convergys
Brian Gallagher
SEI
Ana Maria Guerrero
Jet Propulsion Lab.
Gibbie Hart
SEI
Annabel Kennedy
Jet Propulsion Lab.
Charles Kirby
Jet Propulsion Lab
Paul Nussbaum
TRW
Michael Levesque
Jet Propulsion Lab.
Steven Moore
Raytheon
Brendan O'Connor
Boeing
Bill Wood
SEI

3

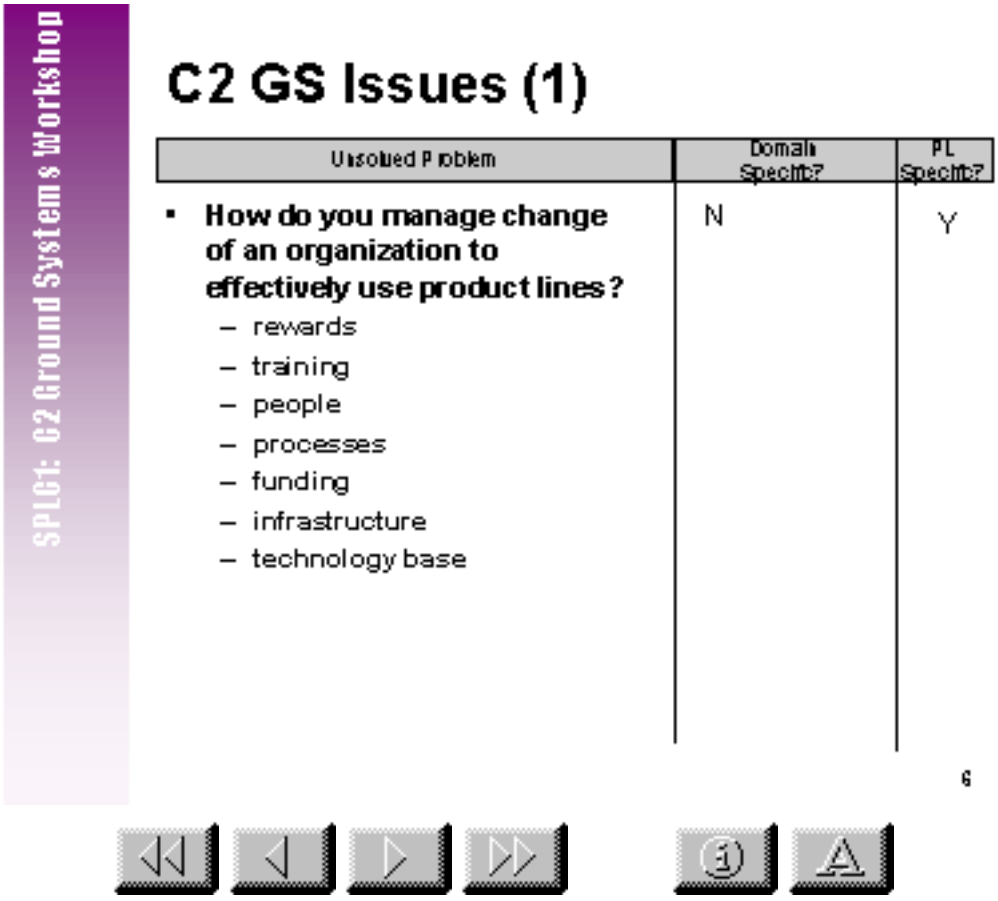



SPLC1: C2 Ground Systems Workshop
Goals of the Workshop
Identify issues that are:
C2 ground system (domain) specific vs. common across multiple domains
product line specific vs. common to all products
Identify recommended actions:
for the PL community
for the command-and-control ground system communities
for academic and industrial researchers
4

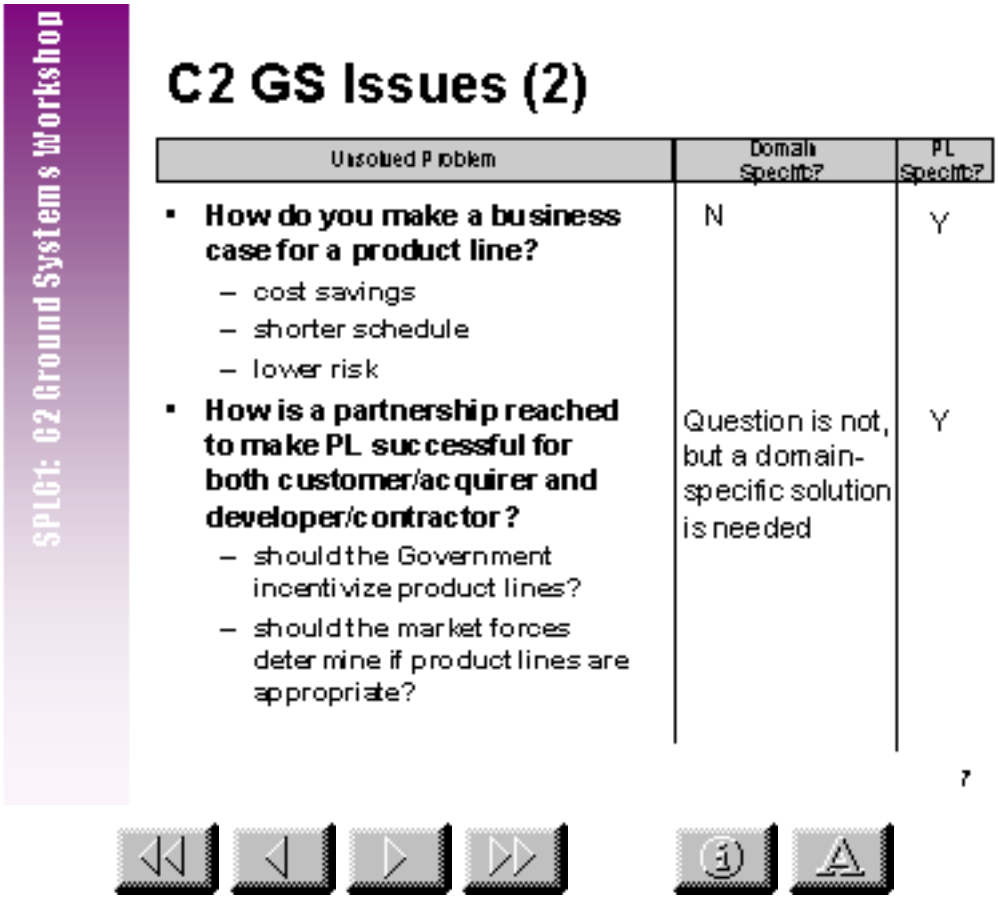
Slide 4 of 10


SPLC1: C2 Ground Systems Workshop
Results of the Workshop
We identified high importance issues in two categories
technical
other
Most of the issues we identified arise in many domains
but we identified the need for C2 specific solutions
5

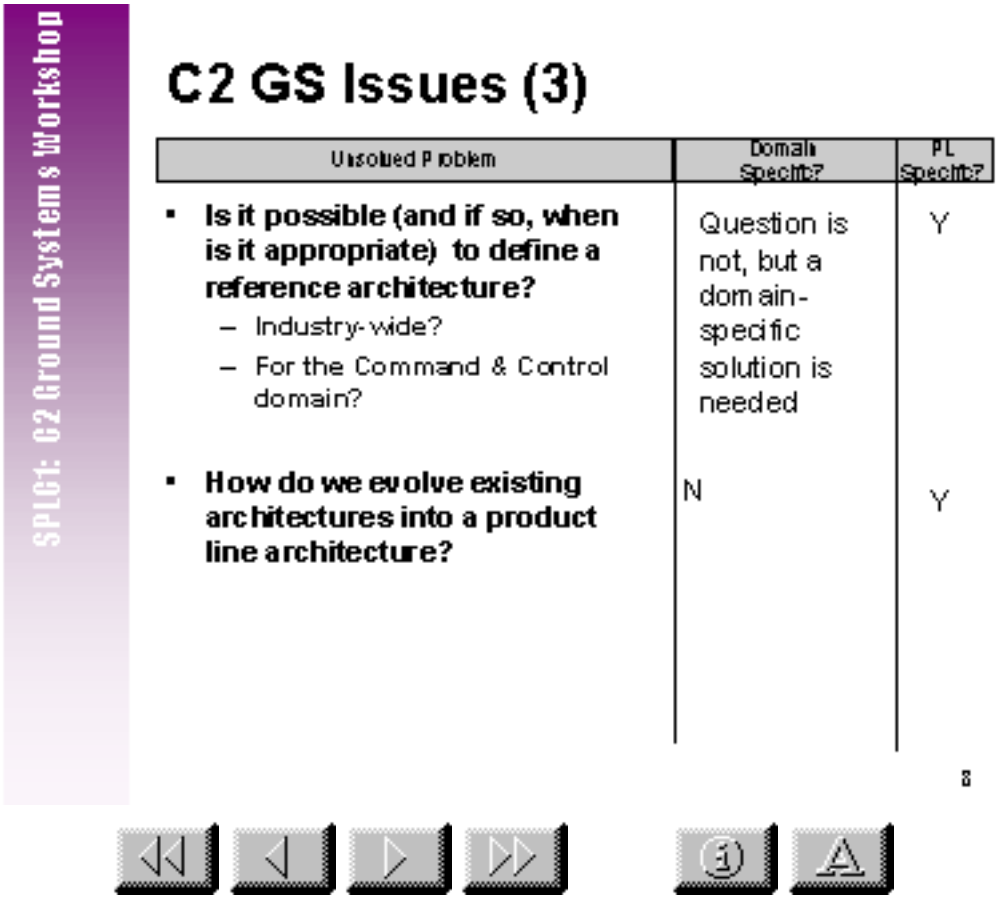
Slide 5 of 10

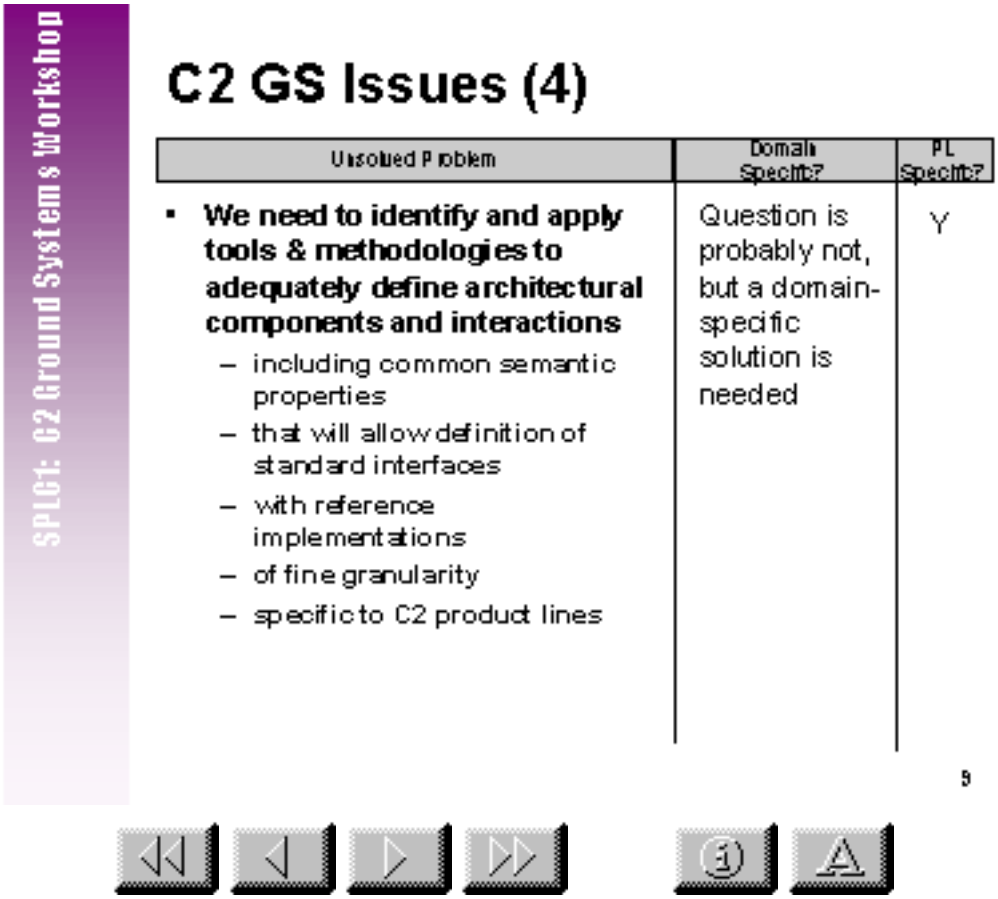

SPLC1: C2 Ground Systems Workshop
C2 GS Issues (1)
Unsolved Problem
Domain Specific?
PL Specific?
How do you manage change of an organization to effectively use product lines?
– rewards
– training
– people
– processes
– funding
– infrastructure
– technology base
N
Y
6

Slide 6 of 10

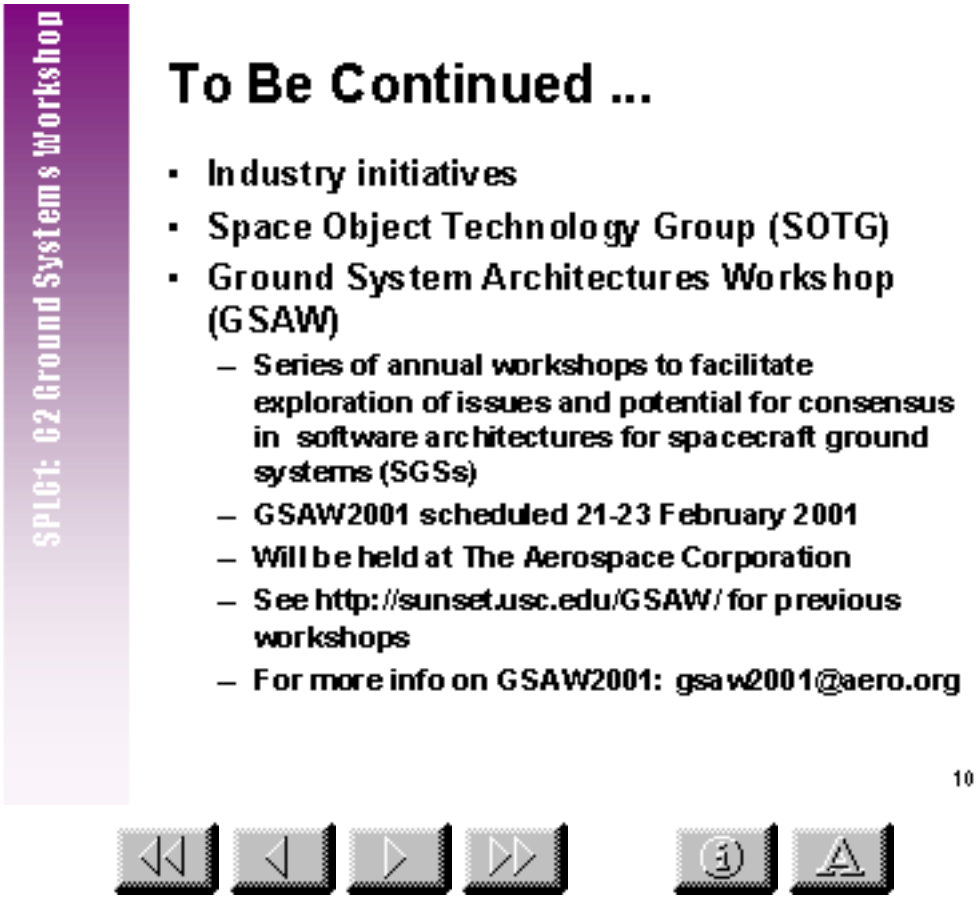

SPLC1: C2 Ground Systems Workshop
C2 GS Issues (2)
Unsolved Problem
Domain Specific?
PL Specific?
How do you make a business case for a product line?
– cost savings
– shorter schedule
– lower risk
N
Y
How is a partnership reached to make PL successful for both customer/acquirer and developer/contractor?
– should the Government incentivize product lines?
– should the market forces determine if product lines are appropriate?
Question is not, but a domain-specific solution is needed
Y
7

Slide 7 of 10


SPLC1: C2 Ground Systems Workshop
C2 GS Issues (3)
Unsolved Problem
Domain Specific?
PL Specific?
Is it possible (and if so, when is it appropriate) to define a reference architecture?
– Industry-wide?
– For the Command & Control domain?
Question is not, but a domain-specific solution is needed
Y
How do we evolve existing architectures into a product line architecture?
N
Y
8


SPLC1: C2 Ground Systems Workshop
C2 GS Issues (4)
Unsolved Problem
Domain Specific?
PL Specific?
We need to identify and apply tools & methodologies to adequately define architectural components and interactions
including common semantic properties
that will allow definition of standard interfaces
with reference implementations
of fine granularity
specific to C2 product lines
Question is probably not, but a domain-specific solution is needed
Y
9

Slide 9 of 10



Slide 10 of 10

Embedded System Product Lines Workshop Results

Presented by Kathy Rose

Slide 1 of 5

Embedded System Product Lines - Workshop Participants

- Bosch - Michael Schlick, Birgit Boss
- Caterpillar - Ron Brown
- Delphi Automotive - David Mannfeld
- General Motors - Kathy Rose
- TRW - Brian Dillard
- Philips Medical Systems - Ben Pronk
- VTT Electronics - Liliana Dobrica

Slide 2 of 5

Embedded System Product Line - Considerations

- Performance (ROM, RAM, Throughput)
- Portability (hardware independent)
- Cost (high volume applications)
- Safety (real-time systems)
- Time to Market

Slide 3 of 5

Embedded System Product Line - Experience

- **Architecture can improve software efficiency**
  - Establish coding standards
  - Eliminate redundant implementations
- **Variants of components have been introduced with reduced functionality**
- **High level language implementations used to improve portability**

Slide 4 of 5

Embedded System Product Line - Discussion

- Need the ability to manage core controller resource usage
- Have difficulty with integration and integration testing the core
- Different approaches on how of core and application resources are allocated
- Migration from core to application back to core (change control and CM issues)

Combining Architecture, Asset Management, and Culture to Successfully Develop Product Lines

The First Software Product Line Conference

(SPLC1)

Denver, Colorado

August 28, 2000

Joe Bauman (joe_bauman@hp.com)

Firmware Architect

Hewlett Packard, Vancouver Division

Phil McCoog (phil_mccoog@hp.com)

Firmware Asset Lead

Hewlett Packard, Vancouver Division

Next slide Back to first slide View graphic version

Workshop Objective

- **Explore the characteristics of Architecture, Asset Management, and Culture & Process**
-
- **Explore how the three can interact to create a sweet spot in support of product line development**
-
- **Examine whether various business requirements change the sweet spot**

Previous slide Next slide Back to first slide View graphic version

Workshop Overview

- **Presentation from each participant on their product line experience**
-
- **Capture elements of each the three domains:**
  - Architecture
  - Asset Management
  - Culture and Process
-
- **Explore interactions between domains**
-
- **Examine current common development models**

Previous slide Next slide Back to first slide View graphic version

Workshop Participant Takeaways

- **There is a universal nature to this problem**
  - DoD, software, embedded firmware
  -
- **Surprisingly, yesterday's battle of componentized architecture is taken as a given**
-
- **Clearer understanding of relatively new domain of asset management**
-
- **Investment balance between three domains will be dynamic**
  - Today's strength is tomorrow's constraint

Previous slide Next slide Back to first slide View graphic version

Workshop Participant Takeaways (cont.)

- **More room, and maybe even need, for democracy than previously thought**
-
- **As product line collaboration expands, massive distribution/diversity has different implications**
  - geographic
  - company
  - culture
  - etc...

Previous slide Back to first slide View graphic version

PPT Slide

Organizers:

J. Jourdan Thomson-CSF

C. Gacek FhG IESE

M. Coriat Thomson-CSF

First Software Product-Line Conference Summary of workshop #2 "Product Line Architecture”
Wednesday, August 30, 2000 Denver, Colorado

Next slide Back to first slide View graphic version

Main focus of PLA workshop

Workshop objective:

- to investigate potential technologies supporting the derivation of architecture variants.

Two main focus:

- Technologies to support variability: Variation points and variability mechanisms (e.g., parameterization, generics, etc.) and criteria for mechanism selection,
- Technologies to support derivation of variants: the decision model (decisions and relationships among them for variant derivation).

Product

Line

Architecture

Previous slide Next slide Back to first slide View graphic version

PPT Slide

Some numbers about the submissions:

- Number of submissions: 18

- Number of reviews per paper: 2 or 3

- European papers: 6

- North American papers:12

- Industry papers: 12

- Academic papers: 6

Information about the reviews

Accepted papers: 11

6 from EU over 6 , 5 from NA over 12

- Is it because the program committee was mostly European or

  is there a more fundamental question behind ?

- Is the American approach of PLA slightly different from

  the European approach of PLA ?

Previous slide Next slide Back to first slide View graphic version

PPT Slide

10 presentations, about 25 attendees

Concepts supporting product line architecture

- Architectural evolution of product families
  A. Maccari
- Using a decision model to support product line architecture modeling, evaluation and instantiation
  O. Flege
- Managing variability in the LCAT SPLIT/DAISY model
  W. El Kaim

Component and computing infrastructure for PLA

- Diversity interfaces, variation and variation management
  F. van der Linden
- Generic software component configuration via partial evaluation
  A. F. Le Meur, C. Consel

Presentations (1/2)

Previous slide Next slide Back to first slide View graphic version

PPT Slide

Presentations (2/2)

Technology supporting PLA development

- Containing and facilitating change via object oriented tailoring techniques
  D. Sharp
- Pattern languages for product line customization
  G. Butler
- Documenting and controlling product lines using the UML
  D. Muthig

Practical experiences on PLA

- An OO application frameworks perspective on commonality and variability in PLA
  H. Hoover, T. Olekshy, G. Froehlich, P. Sorenson
- Addressing variability in a guidance, navigation, and control flight software PL
  D. McComas, S. Leake, M. Stark, M. Morisio, G. Travassos, M. White

Previous slide Next slide Back to first slide View graphic version

PPT Slide

Working groups

We proposed to set up 3 working groups

Suggested working group #1: Variability

Suggested working group #2: Evolution and traceability

Suggested working group #3: Derivation

Previous slide Next slide Back to first slide View graphic version

PPT Slide

Working group #1:Variability Working group #2:Evolution and traceability Working group #3: Other issues

- Various brainstorming and idea production on:
  - PLA and Architecture of a single system
  - Variability

Breakout groups

Previous slide Next slide Back to first slide View graphic version

PPT Slide

Is there really a difference ?

Active discussion on PLA versus Architecture of a single system

- PLA architecture contains
  - the commonalities of several products (it is a part of the architecture of the single systems)
  - plus, in an explicit and documented way, the variabilities of all the products

- Two different trends for the concept of PLA
  - Application oriented PLA yields to a final system
  - Infrastructure oriented PLA aims at facilitating component integration

Previous slide Next slide Back to first slide View graphic version

PPT Slide

- **Different kinds of variability:**
  - required variability: it comes from requirements
  - induced variability: dependency between variability does exist
  - acquired variability: yet acquired variability before building the product line
- **Variability should not be confused with Flexibility:**
  - variability is the difference between several products in the same family,
  - flexibility is related to the capacity of an architecture to extend and satisfy new needs and requirements.

Customisable products and product families are two different things (Some disagreement on this)

Variability versus Flexibility

Previous slide Next slide Back to first slide View graphic version

Variability in architecture

- ## Current Status:
  - Architecture is described using a set of architectural views (no consensus on set of views).
  - Variability is present in all architectural views.
  - In each view, assets can vary:
    - existence, dependency, relationships, lifecycle, static/dynamic, induced/explicit.
  - Variation in assets is represented by several notations (Variation Point, Feature Model, Pattern, Parameters).

- ## Next Step:
  - How to rationalize the selection of the right variability mechanisms to a specific kind of variability in a view ?

Previous slide Next slide Back to first slide View graphic version

Mapping variability between assets

Requirements

functional and non-functional

Variability

- How to map or show impact of variability between assets ?
- How to manage the impact of the variability in the number of traceability links ?

Previous slide Next slide Back to first slide View graphic version

PPT Slide

- **Scope vs Boundaries:**
  - Scope: The scope defines the business and technological limited area taken into account for building a product line
  - Boundaries: Boundaries define the limit where the system wide properties hold. Going beyond such boundaries will break the architecture
  - Guideline: It is good to build product families in a way that the scope can be expanded greatly and still be within the boundaries
- **Context-free design decisions: design decisions that you can combine freely, so that they hold for all components and all systems in the family independent of the context.**
- **Design principle: You can use principles such as "functional decomposition always generates a poor architecture" e.g. to assess a product family architecture.**

  Other statements

Previous slide Next slide Back to first slide View graphic version

PPT Slide

Next workshop

Previous slide Back to first slide View graphic version

WS – Architecture Reconstruction and Product Lines (PL)

Workshop Topics

- **Experiences and techniques for identifying and evaluating commonalities and variabilities**
- **Benefits and costs**
- **Open technical issues**
- **Tool environments**

Next slide Back to first slide View graphic version

WS – Architecture Reconstruction and PL

Relationship between Architecture Reconstruction and PL

- **Product Lines evolve out of existing products**
- **Identification of commonalities and variabilities across existing products**
- **Evaluation of commonalities and variabilities for a) Technical foundation for PL decisions b) Risks, Sensitivity points c) Reference Architectures d) Identification of reuse potential**

Previous slide Next slide Back to first slide View graphic version

WS – Architecture Reconstruction and PL

Discussions:

- **Specific foci for Reconstruction in a Product Line context. Insights:**
  - Hard-wired customer features
  - Hard-wired dependencies to OS/HW/UI
  - Terminology across products
  - Software component granularity
  - Identifying a comparable set of components across products

Previous slide Next slide Back to first slide View graphic version

WS – Architecture Reconstruction and PL

- **What characteristics do PL architectures have?**

  Examples of what styles/structures support specific PL architectural aspects:

  - Isolation of HW (processor, peripherals, …)
  - Communication infrastructure
  - Layering
  - Flexible configuration/calibration
  - Operating system flexibility
  - Virtual devices (indirection)

Previous slide Next slide Back to first slide View graphic version

WS – Architecture Reconstruction and PL

- **Experiences and techniques for identifying and evaluating commonalities and variabilities**
  - Presentation of manual architecture reconstruction in a defibrillator PL context
  - Presentation of MAP (Mining Architectures for PL evaluations) in an automotive PL context

Previous slide Next slide Back to first slide View graphic version

WS – Architecture Reconstruction and PL

- **Tool environments**
  - Overview of the Dali workbench, which supports architecture reconstruction
  - Experiences with the RTM tool in an acquisition environment, which supports requirement traceability

Previous slide Next slide Back to first slide View graphic version

WS – Architecture Reconstruction and PL

- ## Open technical issues
  - Reconstruction to identify feature sets

A software product line is a set of software-intensive

systems sharing a common, managed set of features

that satisfy the specific needs of a particular market

segment or mission

[http://www.sei.cmu.edu/plp/frame_report/what.is.a.PL.htm]

Features

Legacy assets

Reconstruction

Previous slide Back to first slide View graphic version

Measurement Issues for Software Product Line Engineering

Next slide Back to first slide View graphic version

Participants

- Khaled El-Emam - National Research Council, Canada
- Stefan Ferber - Bosch
- Wolf Goethert – SEI
- Cheryl Jones – Practical Software Measurement
- Klaus Schmid - Fraunhofer Institute
- Colin Tully – ESPI
- Dave Zubrow – SEI

Previous slide Next slide Back to first slide View graphic version

Agenda

8:30 Introductions and Review of Workshop Agenda

8:45 Presentations

10:00 Break

Noon Lunch

1:30 Discuss Topics

3:00 Break

4:45 Generate Results Briefing

- Results
- Next steps

5:30 Dismiss

Previous slide Next slide Back to first slide View graphic version

Presentations

- Product Line Indicators and Measures – Dave Zubrow
- Evaluating Component Quality – Khaled El Emam
- Scoping Software Product Lines: Using a Quantitative Approach – Klaus Schmid
-
- Measurement Issues for Software Product Line Engineering – Colin Tully
- Experiment at Robert Bosch GmbH: Case Study of a Product Line for Driver Assistance Systems – Stefan Ferber

Previous slide Next slide Back to first slide View graphic version

Discussion Topics

How Do You Know You Should Adopt a Product Line Approach?

Costs and Benefits of a Software Product Line

Unique Aspects and Challenges for Measurement associated with a Software Product Line

Factors Affecting Implementation of a Software Product Line

Previous slide Next slide Back to first slide View graphic version

Summary - 1

- Many of the measures for product lines are the same as they are for traditional project and organizational management
-
- Potential benefits address a lot of software engineering problems
-
- Regarding the decision to adopt a product line approach, many benefits don’t have associated measures
-
- Both business and technical issues must be addressed - need to distinguish between business and technical measures
-
- Knowing current situation is important as input to transition cost calculations

Previous slide Next slide Back to first slide View graphic version

Summary - 2

- **Total product cost needs allocation from asset development**
-
- **Level of granularity of measures and more views (groupings)**
-
- **Aggregate across multiple products may be misleading, ensuring data consistency**
-

Previous slide Back to first slide View graphic version

SPLC1 Workshop on Product Lines for Command and Control Ground Systems

30 Aug 2000

Next slide Back to first slide View graphic version

Participants (1)

Organizers

Judy Kerner

The Aerospace Corp.

Mark Walker

The Aerospace Corp.

Outbriefer

Mike Hogan

The Aerospace Corp.

Presenters

Don Batory

University of Texas, Austin

Paul Davis

TRW

Susan Kurtik

Jet Propulsion Lab.

Christopher Mushenski

US Army TACOM

John Ohlinger

Nat'l Reconnaissance Off.

Bob Schwanke

Siemens Corp. Res.

Kenneth D. Shere

The Aerospace Corp.

Previous slide Next slide Back to first slide View graphic version

Participants (2)

Lucy Carpenter

Convergys

Brian Gallagher

SEI

Ana Maria Guerrero

Jet Propulsion Lab.

Gibbie Hart

SEI

Annabel Kennedy

Jet Propulsion Lab.

Charles Kirby

Jet Propulsion Lab

Paul Nussbaum

TRW

Michael Levesque

Jet Propulsion Lab.

Steven Moore

Raytheon

Brendan O'Connor

Boeing

Bill Wood

SEI

Previous slide Next slide Back to first slide View graphic version

Goals of the Workshop

- **Identify issues that are:**
  - C2 ground system (domain) specific vs. common across multiple domains
  - product line specific vs. common to all products
- **Identify recommended actions:**
  - for the PL community
  - for the command-and-control ground system communities
  - for academic and industrial researchers

Previous slide Next slide Back to first slide View graphic version

Results of the Workshop

- **We identified high importance issues in two categories**
  - technical
  - other
- **Most of the issues we identified arise in many domains**
  - but we identified the need for C2 specific solutions

Previous slide Next slide Back to first slide View graphic version

C2 GS Issues (1)

- **How do you manage change of an organization to effectively use product lines?**
  - rewards
  - training
  - people
  - processes
  - funding
  - infrastructure
  - technology base

Unsolved Problem

Domain

Specific?

PL

Specific?

N

Y

Previous slide Next slide Back to first slide View graphic version

C2 GS Issues (2)

- **How do you make a business case for a product line?**
  - cost savings
  - shorter schedule
  - lower risk
- **How is a partnership reached to make PL successful for both customer/acquirer and developer/contractor?**
  - should the Government incentivize product lines?
  - should the market forces determine if product lines are appropriate?
  -

Unsolved Problem

Domain

Specific?

PL

Specific?

N

Question is not, but a domain-specific solution is needed

Y

Y

Previous slide Next slide Back to first slide View graphic version

C2 GS Issues (3)

- **Is it possible (and if so, when is it appropriate) to define a reference architecture?**
  - Industry-wide?
  - For the Command & Control domain?
-
- **How do we evolve existing architectures into a product line architecture?**
  -
-

Unsolved Problem

Domain

Specific?

PL

Specific?

N

Y

Y

Question is not, but a domain-specific solution is needed

Previous slide Next slide Back to first slide View graphic version

C2 GS Issues (4)

- **We need to identify and apply tools & methodologies to adequately define architectural components and interactions**
  - including common semantic properties
  - that will allow definition of standard interfaces
  - with reference implementations
  - of fine granularity
  - specific to C2 product lines
  -

Unsolved Problem

Domain

Specific?

PL

Specific?

Y

Question is probably not, but a domain-specific solution is needed

Previous slide Next slide Back to first slide View graphic version

To Be Continued ...

- **Industry initiatives**
- **Space Object Technology Group (SOTG)**
- **Ground System Architectures Workshop (GSAW)**
  - Series of annual workshops to facilitate exploration of issues and potential for consensus in software architectures for spacecraft ground systems (SGSs)
  - GSAW2001 scheduled 21-23 February 2001
  - Will be held at The Aerospace Corporation
  - See http://sunset.usc.edu/GSAW/ for previous workshops
  - For more info on GSAW2001: gsaw2001@aero.org

Previous slide Back to first slide View graphic version

Embedded System Product Lines Workshop Results

Presented by Kathy Rose

Next slide Back to first slide View graphic version

Embedded System Product Lines - Workshop Participants

- **Bosch - Michael Schlick, Birgit Boss**
- **Caterpillar - Ron Brown**
- **Delphi Automotive - David Mannfeld**
- **General Motors - Kathy Rose**
- **TRW - Brian Dillard**
- **Philips Medical Systems - Ben Pronk**
- **VTT Electronics - Liliana Dobrica**

Previous slide Next slide Back to first slide View graphic version

Embedded System Product Line - Considerations

- Performance (ROM, RAM, Throughput)
- Portability (hardware independent)
- Cost (high volume applications)
- Safety (real-time systems)
- Time to Market

Previous slide Next slide Back to first slide View graphic version

Embedded System Product Line - Experience

- **Architecture can improve software efficiency**
  - Establish coding standards
  - Eliminate redundant implementations
- **Variants of components have been introduced with reduced functionality**
- **High level language implementations used to improve portability**

Previous slide Next slide Back to first slide View graphic version

Embedded System Product Line - Discussion

- Need the ability to manage core controller resource usage
- Have difficulty with integration and integration testing the core
- Different approaches on how of core and application resources are allocated
- Migration from core to application back to core (change control and CM issues)

Previous slide Back to first slide View graphic version