

REUSABLE SOFTWARE COMPONENTS

Trudy Levine
Fairleigh Dickinson University
Teaneck, NJ 07666
levine@fd.edu
http://alpha.fdu.edu/~levine/reuse_course/columns

What are reusable software components? Software that has been reused includes source code, requirements analysis, domain analysis, system designs, user manuals, test suites, and other types of documented system knowledge that has been reapplied to other applications (generality), systems (portability) or time periods (maintenance). Software reuse has been common since the days of sharing an executing sine routine in octal on punched cards. Its benefits include assisting in the management of complexity, reliability and cost.

The Ada Language is particularly appropriate for supporting reuse, since it provides generic programming, object-oriented programming, standardization, and machine independence, among many other features that assist in maintenance and generality. The self-documenting features of Ada source code, specifically, assist the programmer in maintaining rather than reinventing software.

This column is about courses in software reuse. Their descriptions, obtained from the sponsoring organizations, list the issues involved in reuse. No endorsement of any of these courses is implied.

AdaWorks

For the past ten years, AdaWorks has specialized in software development, consulting, and training for the Ada software industry. The Ada software applications experience includes embedded systems, information systems, and scientific computing. Training courses cover a full range of software engineering methods. Programming expertise, in addition to Ada, includes object-oriented programming languages, object-based languages, and procedural languages.

Workshop in Software Reuse 3 days: (Recommended: an Introductory Ada, C++, or Java class)

This course is appropriate for students who have completed an introductory or intensive class in Ada or have experience with C++. Students complete a small project using reusable software components. Topics include: what can be reused, experimenting with components, management of components, buy versus build decision, the "container" class, reuse versus "leveraging", Ada generic components (including generic formal subprogram parameters) and C++ templates, reuse with different design methods (OOD, SASD, LVM, etc.) Management issues related to software reuse are examined.

Contact: Richard Riehle/Sera Hirasuna
6 Supulveda Circle
Salinas, CA 93906
Phone: (831) 443-5536
Email: info@adaworks.com
URL: <http://www.adaworks.com/courses.asp#reuse>

SIGAda

See the SigAda education site for a list of Ada courses, specifically those related to software engineering and object-oriented design.

www.acm.org/sigs/sigada/education

THE ASE2 CDROM

Course on Software Reuse: SE 508

The purpose of this course is to explore contemporary topics in systematic software reuse. This includes the impact of Object-Based and Object-Oriented Design and Programming with Ada83, Ada95, and C++ along with Domain Engineering on the software development process. The course concentrates on the practical aspects of applying architecture-centric, domain-specific, library-based reuse methodologies integrated with the software development process to create software systems in an efficient, cost-effective manner. The course illustrates how object-oriented and domain engineering techniques coupled with domain-specific libraries can be used to effectively develop significant software systems in a short period of time, frequently realizing reuse on the order of 70% or more. Libraries of object-based reusable software components will be used to design and implement solutions to problems.

Material presented in this course includes information from the Software Productivity Consortium (the Reuse-Driven Software Processes Guidebook), the Air Force's Comprehensive Approach to Reusable Defense Software program, ARPA's Software Technology for Adaptable Reliable Systems program, Europe's ESPRIT III Project #7808 (REBOOT - Reuse Based on Object-Oriented Techniques - see below), and several other sources (including IBM and HP). All of the reading material can be found in the Public Ada Library.

The major sections of the course include:

- * Review of Software Engineering and How Reuse Fits In
- * Object-Based and Object-Oriented Analysis and Design (with emphasis on designing with reuse, comparing and contrasting Ada83, Ada95, C++)
- * Domain Engineering
- * Designing for Reuse

Richard Conn also teaches a classroom course, CSIS 1020 (<http://unicoi.kennesaw.edu/~rconn>), with a heavy reuse and software engineering orientation, and discusses reuse in Microsoft technologies (which supports a \$3B/year third-party reuse industry.)

For information

CONTACT: Richard Conn

rickconn7@msn.com

~~~~~

## APPRAISAL

The APPRAISAL project is a technology transfer project that is intended to support the industrial exploitation of software reuse. It is partially funded by the European Commission under the ESSI Initiative.

Information on software reuse in general, and about software reuse with Ada in particular, can be found at:

[http://www.comp.lancs.ac.uk/computing/research/cseg/projects/APPRAISAL/ada\\_course.html](http://www.comp.lancs.ac.uk/computing/research/cseg/projects/APPRAISAL/ada_course.html)

[http://www.comp.lancs.ac.uk/computing/research/cseg/projects/APPRAISAL/reuse\\_course.html](http://www.comp.lancs.ac.uk/computing/research/cseg/projects/APPRAISAL/reuse_course.html)

~~~~~

Carnegie Mellon Software Engineering Institute

Carnegie Mellon Software Engineering Institute offers a videotaped course on *Developing Reusable Software*. The course describes reuse-driven development processes as well as analysis and design methods that promote reuse. The course uses comparisons and examples to present alternative approaches for domain analysis, domain design, component implementation, and components-based applications engineering.

By taking this course, participants will gain an understanding of what makes a software component more or less reusable, and what is necessary to reverse engineer existing software that has reuse potential but that is not entirely reusable as it stands. Those responsible for implementing a software reuse initiative will be able to define technical activities of design with reuse and design for reuse, and assess

linguistic and methodological support. The course also provides knowledge necessary for populating a library with well-designed components that can be reused by other people and projects.

The course consists of lectures and problem-solving activities, with a strong emphasis on laboratory work, and can be used as direct delivery or as a supplement to existing related courses. Lecture material is mostly language independent; however, component development and modification are required to illustrate the notions of reusability and reengineering. The Ada language is designed to support the goals and principles of modern software engineering, particularly those of software reuse, and is used during the course as the primary standard notation for illustration and development purposes.

Topics include:

- * The Reuse Process
- * Characteristics of Reusable Software
- * Domain Analysis
 - * Object-oriented models
 - * Feature-oriented domain analysis
- * Domain Design
 - * Domain-specific software architectures
 - * Object-oriented design
- * Domain Implementation
 - * Component specification
 - * Component Design
- * Software construction with reusable components

<http://www.sei.cmu.edu/products/videos/dev.reusable.sw.html>

REUSE EDUCATION IN FRANCE

Software REUSE a Holistic Approach : A two-day course embracing business, organizational, engineering, and resource management across the entire software process.

AUDIENCE : The course is intended for managers and developers from an organisation that aim for competitive advantages in software development.

PURPOSE : Software reuse will, according to Barry Boehm at DARPA, be one of the major sources of saving in software development over the next 15-20 years. By reusing systems or system parts that already have been developed, an organization enhances its possibilities to both improve the productivity and the quality of the produced software.

The course is based on the work performed within REBOOT (Reuse Based on Object-Oriented Techniques) and SER (Software Evolution and Reuse), two major projects in the reuse-area in the ESPRIT programmes. The REBOOT project has covered all the major aspects of reuse and industrial experience has been obtained. SER is a follow up project aimed at disseminating the knowledge from REBOOT and several other ESPRIT projects. The course is a complete reuse methodology (REBOOT) and the result from thirteen real life software development projects adapted for reuse of large and generic components with a focus on object-oriented frameworks and patterns.

COURSE OUTLINE:

Overview: Introduction to the important aspects of a mature reuse organization. Examples of successful reuse experiences from different domains are presented.

Organizing reuse : Strategic advantages and challenges in changing to organized reuse. Presentation of different mature reuse organizations based on different market and product scenarios. New and changed roles necessary in a reuse organization.

Managing reuse projects : Reuse from the project managers viewpoint. Topics discussed:

- * different types of reuse activities in a development project and how to manage them,
- * new roles involved in a reuse project, their potential conflicting interests, and how to handle them
- * changes in project planning, managing and following up when organized reuse is involved.

DEVELOPMENT FOR REUSE : An overview of the specific problems connected to the development of reusable components including domain analysis and guidelines for design and implementation.

Re-engineering for reuse : Re-engineering by using semi-automatic techniques for restructuring the existing software so it can be reused.

Component management: How to effectively store and retrieve reusable components in a mature reuse organization. Issues covered:

- * additional information necessary to make a component reusable and to continuously improve its reusability,
- * classification of components
- * configuration management of components
- * management of the component library

REUSE METRICS : Description of how to measure if reuse is having the desired effect. Main focus on:

- * Process metrics, i.e. how to measure quality, lead-time and productivity in the presence of reuse.
- * Product metrics, i.e. how to measure the reusability and quality of reusable components.

CASE STUDIES AND EXAMPLES: Presentation of projects and organisations that have introduced software reuse. Covering both organization aspects as well as technical challenges.

REUSE INTRODUCTION PROCESS : A detailed description of the Reuse Maturity Model which is used to determine the organizations current reuse maturity, as well as giving guidance on how to introduce reuse. Also the Reuse introduction plan, which is a template for how to plan, execute and monitor a reuse introduction programme.

ADAPTING EXISTING DEVELOPMENT PROCESSES TO REUSE: Outline of how to adapt the existing development process of a mature organization to reuse, without changing entirely to a pure object-oriented development process.

REBOOT REUSE TOOLS: Description of prototype tools developed within the REBOOT project to support parts of the reuse process, and how they have been incorporated in commercial CASE tools. Mainly support for classification, product metrics and re-engineering.

FRAMEWORKS AND PATTERNS: Frameworks are generic components developed and reused for applications within a domain. Patterns are good design solutions solving a certain design problem that can be reused itself or used as a building block of a framework.

CONTACT: Jean-Marc MOREL PC: CL F12D22
 Software Development Methodology
 Bull S.A
 Rue Jean-Jaures
 F-78340 Les Clayes-Sous-Bois
 FRANCE

E-mail : Jean-Marc.Morel@bull.net
Phone: +33 (0)1 3080 7448 Fax: +33 (0)1 3080 7078

~~~~~

## Ohio State University

The Ohio State University (OSU) offers an integrated year-long sequence of software component engineering courses within its undergraduate computer science curriculum. The sequence begins with the first course for Computer Science majors (but for which some prior programming experience at the level of a high school course is a prerequisite). The sequence builds on the work of the OSU Reusable Software Research Group. Several years of experience and evaluation of this approach using RESOLVE/C++ has demonstrated that even freshmen are able to understand RESOLVE-style formal specifications and are able to use them to reason about the behavior of component-based systems.

A Software Composition Workbench, consisting of an integrated set of software tools, supports the RESOLVE approach to component-based software development and is under active development. The workbench guides students in their use and application of RESOLVE methodology with C++ as the delivery vehicle.

Materials used in the courses in RESOLVE/C++ notation are available for dissemination. Tools of the workbench are expected to be released in the future.

Contact:   Bruce W. Weide, E-mail: weide@cis.ohio-state.edu  
              RSRG Web page: <http://www.cis.ohio-state.edu/rsrg/>  
              Course Sequence Web page: <http://www.cis.ohio-state.edu/~weide/sce/now>

~~~~~