## Demonstration Schedule

**Tuesday, February 9, 1999**  
**Time: 12:00pm-2:00pm**

<table>
<thead>
<tr>
<th>Demo name</th>
<th>Demoer</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architect's Automated Assistant</td>
<td>Alex Aegy</td>
<td>SAL 327</td>
</tr>
<tr>
<td>COCOMO II.1999</td>
<td>Jongmoon Park</td>
<td>SAL 318</td>
</tr>
<tr>
<td>COCOMO 'I.1999 Spreadsheet Version</td>
<td>Winsor Brown</td>
<td>SAL 326</td>
</tr>
<tr>
<td>COPROMO</td>
<td>Winsor Brown</td>
<td>SAL 326</td>
</tr>
<tr>
<td>CORADMO</td>
<td>Cyrus Fakharzadeh</td>
<td>SAL 337</td>
</tr>
<tr>
<td>Digital Library Application Prototype -- Hispanic Archive</td>
<td>Nikunj Mehta</td>
<td>SAL 324</td>
</tr>
<tr>
<td>Digital Library Application Prototype -- World War I Archive</td>
<td>Rajnish Lal</td>
<td>SAL 324</td>
</tr>
<tr>
<td>Distributed Collaboration and Prioritization Tool</td>
<td>Jungwon Park</td>
<td>SAL 329</td>
</tr>
<tr>
<td>DRADEL Architecture Tool</td>
<td>Neno Medvidovich</td>
<td>SAL 336</td>
</tr>
<tr>
<td>Process Model Decision Assistant</td>
<td>Chris Abts</td>
<td>SAL 330</td>
</tr>
<tr>
<td>Protocol Generation</td>
<td>JuneSup Lee</td>
<td>SAL 320</td>
</tr>
<tr>
<td>SCOST</td>
<td>Hoh In</td>
<td>SAL 322</td>
</tr>
<tr>
<td>WinWin--Defense Application (Unix)</td>
<td>EDCS Team: Judy Kerner and Dan Port</td>
<td>SAL 324</td>
</tr>
<tr>
<td>WinWin--Digital Library Application (Windows)</td>
<td>577a Team: Erin Ballew, Nikunj Mehta, and Anne Lynch</td>
<td>SAL 324</td>
</tr>
</tbody>
</table>

*(Please see posted floor plans for demonstration locations.)*
Name: AAA (Architect’s Automated Assistant)

Objective: To identify potential architectural mismatches between high level (COTS) software components of software systems.

Rationale: Since, composing software systems out of COTS components is becoming more popular, the risks of undesirable side effects these components might have onto other components are becoming increasingly severe. Although COTS components may run perfectly on their own, they may not work anymore when used together. The AAA tool assists in identifying potential problems early on to avoid software composition problems later on.

Target Users: This tool/model is primarily meant for software developers for risk assessment during the early stages of the software life-cycle when only little is known about the ‘interior’ of major software (COTS) components.

Scope: AAA provides an environment in which architects and software designer can specify a system in terms of components and their properties. Based on those properties, the tool can then suggest potential problems which should be investigated prior to committing onto any COTS product.

Project Type: The model is the cumulative result of the dissertation work of Cristina Gacek and Ahmed Abd-Allah. Both projects were also funded in part by DARPA and Rome Labs.

Developers: The tool was developed as part of Gacek’s dissertation in 1998.

Runs On: Since AAA is implemented in Java it runs on all platforms which support that language. It was tested under Windows NT and Solaris.

IPR Status: AAA copyright owned by CSE.

Technical Approach: N/A

Future Directions: N/A
Demonstration Guide - USC-CSE Annual Research Review 1999

Name: COCOMOII.1999.0 (COnstructive COst Model)

Objective: To develop software cost and schedule estimation model tuned to the life cycle practices of the 1990's and 2000's; to develop software cost database and tool support capabilities for continuous model improvement; to provide a quantitative analytic framework, and set of tools and techniques for evaluating the effects of software technology improvements on software life cycle costs and schedule.

Rationale: Tailorability of an organization process to its own process drivers; consistency of the granularity of the cost estimation with the granularity of available information; provide range estimates to the degree of definition of the estimation inputs.

Target Users: Customers, Managers, System Engineers, Software Engineers, Cost Analysts

Scope: Generation of Effort and Schedule Estimates; Calibration; Risk Assessment

Project Type: Multi-year USC-CSE research project

Developers: Principal Investigator: Dr. Ellis Horowitz; Student Programmers and Testers: Jongmoon Baik

Runs On: SunOS 4.x and 5.x, Windows95/NT

IPR Status: COCOMOII copyright owned by USC-CSE. Affiliates free to use, modify, but not restrict other affiliates' use

Technical Approach: COCOMOII follows the openness principles used in the original COCOMO. Thus, all of its relationships and algorithms will be publicly available. Also, all of interfaces are designed to be public, well-defined, and parameterized, so that complementary preprocessors, post-processors, and higher level packages, can be combined straightforwardly with COCOMOII.

Future Directions:
Integrate COTS Integration cost model(COCOTS)
Publish COCOMOII book with USC COCOMOII CD
Name: Spreadsheet COCOMO II

Objective: provide a version of COCOMO II that incorporates all the driver value selection criteria and makes the model equations visible, thus assisting students in understanding how to use COCOMO II.

Rationale: The existing COCOMO II tools allow the selection of driver levels but provide only descriptions of the driver and its level selection criteria via help. This implementation integrates the selection criteria and supporting arithmetic calculations with the selection.

Target Users: Students or new COCOMO II users.

Scope: Single module projects with source lines of code input and Post-Architecture or Early Design models. All the COCOMO II drivers are included.

Project Type: USC graduate student directed study project.

Developers: Ms. Catherine Kao and A. Winsor Brown.

Runs On: Excel Office 97 version; version 4(?) or higher upon request and at own risk (multiple worksheet; no macros).

IPR Status: Copyright USC-CSE.


Future Directions: Possible extension to multiple modules per project. Possible export/generation of .est files. No intention of expanding to cover maintenance effort, local calibration, or phase/stage distribution.
**Name:** Constructive Productivity Model - COPROMO

**Objective:** Provide strategic planning decision assistant for senior management.

**Rationale:** Existing parametric models, COCOMO-II and CORADMO, are used as evaluation tools for projected costs of development of a prototypical application from the domain of interest.

**Target Users:** Senior management and staff, planners and SEPGs.

**Scope:** A broad application of the COCOMO related models to assess impacts of driver changes into the future in order to make trade-offs of investment strategies in technology/process.

**Project Type:** multi-year USC-CSE research project, initially supported by and implemented for KBSA Life Cycle Evaluation, AFRL.

**Developers:** A. Winsor Brown and Dr. Barry Boehm.

**Runs On:** Excel Office 97 version; version 4(?) or higher upon request and at own risk (multiple worksheet; no macros).

**IPR Status:** Copyright USC-CSE.

**Technical Approach:** The model is based on the use of COCOMO II\(^1\) and CORADMO\(^2\) as valuation mechanisms. The implementation approach uses a representative application from the domain of concern to the senior management, and the identification of technology drivers and time frames. One version of the tool, a Technology Impact Analyzer, has been implemented and used in the evaluation the Knowledge Based Software Assistant.

The COPROMO implementation approach is to identify an application, time frames and specific technologies that are expected to impact productivity for the prototypical application over the time frames selected. The prototypical application should be one that is representative of the domain of concern of the senior management. The time frames should be long enough to have the selected technology mature and come into use, spanning at least eight to fifteen years. The specific technologies should be identifiable and have relatively clearly defined, even if still evolving, content. One of the technologies should be the commercial and milieu specific (e.g. DoD) technologies that will evolve independently of the specific technologies.

---

\(^1\) Constructive Cost Model, version II, 1998 calibration.

\(^2\) Constructive RAD-schedule Model, a currently uncalibrated extension to COCOMO II.
The parametric drivers of the valuation model include COCOMO II’s effort scale factors and multipliers and which cover process, product, platform, personnel and project; and CORADMO’s schedule and effort multipliers. Each of the drivers’ values are then gathered for the current baseline or assessed into the future using engineering judgement based on the assumed impacts of the selected, specific technologies.

All of the information on the drivers, their evolution over time, and the rationales are then input into a spreadsheet tool. The tool, called a Technology Impact Analyzer, consists of multiple, parallel COCOMO II and CORADMO parametric model executions. The tool graphically displays each of the drivers’ values over time to allow reasoning and discourse about their values and evolution over time. The tool also provides fields for the capture of the rationales for each of the drivers’ values and evolution on the same page as the tabular and graphic display of values. Finally, the tool displays a comprehensive set of graphs showing the impact of the selected technologies over time for the issues of concern: effort, schedule and recommended head-count.

**Future Directions:** KBSA Technical Impact Assessment Tool use and feedback.
Generation of more example implementations to get more experience. Future extensions possible into investment cost projections/calculations and ROI.
Name: CORADMO

Objective: The intent of the CoRADMo.xls is to calculate/predict the schedule (months, M), personnel (P), and adjusted effort (person-months, PM) based on the distribution of effort and schedule to the various stages, and impacts of the selected schedule driver ratings on the M, P, and PM of each stage.

Rationale: The COCOMO RAD model is an extension of the COCOMO II model, which focuses on the cost of developing software using rapid application development techniques. RAD is taken to mean an application of any of a number of techniques or strategies to reduce software development cycle time.

Target Users: Managers, Project Planners, and Software Engineers

Scope: CoRADMo.xls takes its data from the exported output of a COCOMOII.exe run via an intermediate tool, CoCoMoII.xls that also provides useful results itself.

Project Type: Multi-year USC-CSE research project

Developers: A. Winsor Brown and Cyrus Fakharzadeh

Runs On: Microsoft Windows 95/98 and NT 4.0 with Excel 97

IPR Status: Copyright USC-CSE.

Technical Approach: Users need to be proficient with the use of Microsoft Excel.

Future Directions: To eventually have a Windows/Java program that performs the same tasks as this spreadsheet with the added functionality of the USC-COCOMO.exe.
Demonstration Guide - USC-CSE Annual Research Review 1999

Name: Hispanic Digital Archive

Objective: Demonstration of prototypes built for Library projects in the Software Engineering Class CS 577a.

Rationale: Library projects follow an MBASE process and result in creation of elaborate prototypes to resolve key risks early. Prototypes are demonstrated to the customer and users and lead to consensus and common understanding of requirements.

Target Users: Library personnel including Archive Curators, social sciences researchers and general public.

Scope: To construct a global interface for easy access to rare archived materials. Scope includes building both end-user and administrative interfaces to USC - ISD database backend based on IBM's Digital Library and transitioning the software to its users.

Project Type: The project is a result of collaboration between USC-CSE and ISD running Fall 1998 through Spring 1999.

Developers: Erin Ballew; Sheelu Bhansali; Cyrus Fakharzadeh; Nikunj Mehta; Didi Yao; Juan Carlos Diaz

Runs On: Web interface runs on any platform with Internet Explorer 4.x and Netscape Navigator 4.06+. Administrative interface on PCs

IPR Status: Copyright USC-CSE.


Future Directions: Reuse opportunities for University Digital Archive based on key lessons learned from project.
Name: World War I Book Collection – Access Enhancement

Objective: To provide detailed information accessible through the Internet about the World War I book collection.

Rationale: The World War I book collection is in the process of being shifted to remote storage. Books stored remotely cannot be viewed before being retrieved. The World War I Book Collection – Access Enhancement project aims to provide a solution to this problem by allowing users to search and view detailed information, such as, the Table of Contents, Abstract, maps, etc. of these books.

Target Users: Students and Faculty of the History & Political Science department and people interested / researching World War I books.

Scope: The project will involve displaying scanned images of the Table of Contents, Abstract, Maps, etc. of these books. It will also provide keyword based search capability for the Title, Author, Abstract, Table of Contents, etc.

Project Type: This is a two – semester project, the analysis and design phase was done as part of 577a in Fall 98. The construction phase is being carried out as part of 577b this semester (Spring 99).

Developers: Project being developed as part of CSci 577 - a course on Software Engineering taught by Professor Barry W. Boehm. The team developing the project includes Rajnish (Manager), Chetan (Architect), Takashi, Amar and John.

Runs On: Windows NT, 95 & 98, Sun Solaris 5.06, Macintosh and other systems that support Netscape 3.0 or higher / Internet Explorer 4.0 or higher.

IPR Status: Project is owned by the USC – CSE and is part of the CSE Archives. The data and information about books will belong to the VKC Library, which owns the World War I book collection.

Technical Approach: This project is being developed based on the MBASE guidelines using various tools for software development. The application of the various tools will be demonstrated during the Demo. The tools include WINWIN, USC COCOMO II, Rational Rose, project planners, etc.

Future Directions: Can be considered for other collections that need to be stored remotely, or online snap shots of non-book material.

Demo Description: none
Name: Distributed Collaboration Priorities Tool (DCPT)

Objective: Distributed Collaboration Priorities Tool (DCPT) assists in the prioritization of software development items such as requirements, goals, and Win-conditions.

Rationale: One of the most common problems in collaborative development task is how to prioritize. In particular requirements have proven elusive in this since software developers cannot implement stakeholders’ requirements fully when time and resources are limited. To solve the problem, requirement engineers must prioritize requirements. The problem is exacerbated when the critical stakeholders are not all in the same location and/or can not collaborate at the same time. Under such diverse conditions, DCPT can aid in the collaborative prioritization of requirements and other project critical items and tasks.

Target Users: All stakeholders in the front end of the software development process: requirement engineers, customers, developers, representatives of users.

Scope: DCPT is used in the prioritization of development items and tasks. Users assign each item a difficulty and importance for which summary statistics are generated and used to place the items into priority “bins” that classify them into relative prioritization’s.

Project Type: USC-CSE SEDA research project.

Developers: Dr. Barry Boehm, Dr. Dan Port, and Jung-Won Park.

Runs On: Any computer supporting Java 1.1.6 or higher version.

IPR Status: Copyright owned by USC-CSE.

Technical Approach: If the number of requirements or of stakeholders increases, it is difficult to prioritize requirements collaboratively especially when the stakeholders are distributed. DCPT aids in the collaborative prioritization of development items. DCPT supports three different priority “bin” models categorization: relative average bin model, equal areas bin models, and ratio range bin model. Each model can be converted to the other easily.

Future Directions: Distributed prioritization, Disambiguation of item priorities, Determination of item interdependencies, Integration with WinWin negotiation tool.
Name: DRADEL

Objective: An extensible environment for architecture-based software development and evolution.

Rationale: Software architectures are high-level models of software systems. Software evolution techniques typically available in programming languages (e.g., subtyping) may be inadequate and/or overly restrictive at the level of architecture. Furthermore, architecture-level properties of a system are of little value unless they can be transferred to the systems implementation in a systematic manner.

Target Users: The typical users are software architects (for constructing architectural models) and developers (for generating software system skeletons from architectural descriptions).

Scope: DRADEL provides architecture modeling, analysis, implementation, and evolution support at system specification time. It is extensible to provide such support at execution time.

Project Type: Multi-year UC Irvine research project to be continued as a USC-CSE project, currently supported by DARPA.

Developers: Primary developer is Neno Medvidovic. The implementation infrastructure upon which DRADEL is built was developed by Peyman Oreizy and Neno Medvidovic. Richard Taylor and David Rosenblum have participated in developing DRADEL’s conceptual architecture.

Runs On: Java

IPR Status: Copyright owned by the University of California.

Technical Approach: DRADEL was constructed to support architectures built and evolved according to the rules of the C2 architectural style. Just like the application architectures it is built to support, DRADEL itself adheres to C2 style rules: it has a tiered architecture comprising several components, which communicate solely via flexible connectors. The architecture of the DRADEL environment has been designed to be easily evolvable, so that new components may be added or existing components replaced to satisfy new requirements. DRADEL itself can be used reflexively to model and ensure the consistency of its own evolution, i.e., DRADEL’s architecture can be specified in its supported ADL, parsed, analyzed, and the environment itself partially generated using DRADEL.

Future Directions: Integration with execution-time software evolution tools. Automated off-the-shelf component retrieval and population of partial architectures. Implementation of optimized software connectors. Investigation of the applicability of the tool integration framework to other architectural styles/approaches.
Name: S-COST II (Software Cost Option Strategy Tool II)

Objective: This exploratory knowledge-based tool is for assisting stakeholders to surface appropriate cost-resolution options, to visualize the options, and to negotiate a mutually satisfactory balance of requirements and cost.

Rationale: A critical success factor in requirements engineering is the balance of requirements commitments with available cost resources. Many software systems have come to grief due to a combination of commitments to costly but low-utility requirements (gold-plating), and/or inadequate budgets for high-utility requirements.

Target Users: Initially all participants in the front end of the software process: system engineers, representatives of users, customers, developers, maintainers.

Scope: Option generation and negotiation aids for top-level cost and quality conflicts.

Project Type: Multi-year USC-CSE research project.

Developers: Barry Boehm and Hoh In

 Runs On: Sun hardware platform running X11 server; no other restrictions.

 IPR Status: S-COST copyright owned by USC-CSE.

Technical Approach: S-COST examines the COCOMO cost drivers and Win Conditions and provides draft cost Options using the Cost Resolution Strategies (e.g., Reduce/defer functionality, Reduce/defer quality, Improve tools, techniques, platform, personnel capability, Relax schedule constraint, Reuse software assets, and Increase budget) with visualization aids. It operates in the context of the USC-CSE WinWin system, COCOMO, and QARCC.

Future Directions: Development of more detailed cost-related option generation and resolution capabilities.

URL Reference: http://sunset.usc.edu/~hohin/scost
Name: Using WinWin to get Agreement on Software Requirements

Objective: Extending the USC WinWin System to include architecture-level decision rationale capture to support continuous evolutionary development of families of long-lived software-intensive systems.

Rationale: The current paradigm has shifted to incremental evolution of systems. It is extremely important to involve critical stakeholders in the decision making process, capture multiple stakeholders perspectives and their rationale, provide tools to analyze and evaluate architecture design alternatives and record how the WinWin solutions are achieved.

Target Users: Initially all participants in the front end of the software process: system engineers, representatives of users, customers, developers, maintainers.

Scope: WinWin provides an environment in which stakeholders can define and refine win conditions for the next step of the Spiral Model. There is support for collaborative product and process definition and negotiation. The environment offers tools for analyzing conflicting win conditions.

Project Type: Multi-year USC-CSE research project. Supported by DARPA and USC-CSE Affiliates.

Developers: Development team includes Dr. Barry Boehm, Dr. Anne Curran, Dr. Ellis Horowitz, Dr. Hoh In, June Sup Lee, and Dr. Ming June Lee.

Runs On: Sun/HP with Unix, X11 server and Motif library, PC running Linux with Motif library, and any computer supporting Netscape 3.0 or higher for running the Java version. NT version under development.

IPR Status: WinWin copyright owned by USC-CSE

Technical Approach: The WinWin support system has a distributed architecture. A stakeholder interacts with a WinWin client that communicates to other WinWin clients via the WinWin DB-server. The DB-server maintains information on the stakeholders involved in a project and their corresponding WinWin database locations. A set of tools are directly integrated with the WinWin - for example the ACT for architecture-level design rationale capture, the COCOMO tool for cost analysis, the Netscape and Framemaker tools for documentation and printing. Stakeholders can also attach their favorite tools and associated files to provide annotation, rationale or analysis for an artifact. The support environment provides a basis set of operations on the WinWin artifacts (win conditions, issues, options and agreements) such as relating, commenting, state propagation, and messages.

Future Directions: Process agenda. Conflict and option advisor. Integration with other software engineering tools.