

Adoption Centric Software Engineering

- ❑ Introduction
- ❑ ACRE Live Documents
- ❑ ACRENotes
- ❑ ACRE Persistence Engine
- ❑ ACRE Visualization Engine
- ❑ ACRE Scenarios
- ❑ www.cs.uvic.ca, acse@cs.uvic.ca

CASCON 2002

Hausi A. Müller, Margaret-Anne Storey, Anke Weber, Will Kastelic, Holger Kienle, Qin Zhu, Jun Ma, Fang Yang, David Zwiers, University of Victoria, Canada

Ken Wong, University of Alberta, Canada

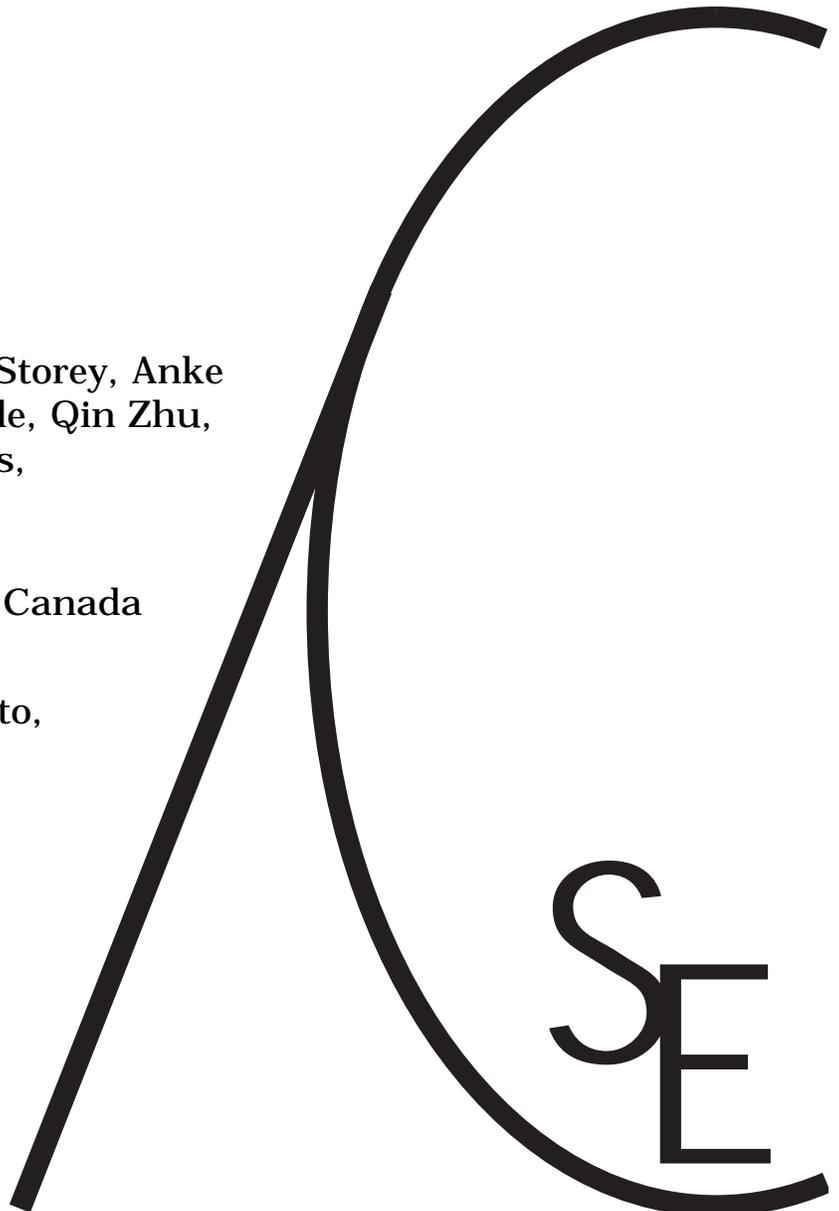
Jon Pipitone, University of Toronto, Canada

Sponsored by

NSERC

IBM

CSER





Leveraging Cognitive Support and Modern Platforms for Adoption-Centric Reverse Engineering (ACRE)

Hausi A. Müller, University of Victoria, Canada [hausi@cs.uvic.ca]

Problem

Research tools in software engineering often fail to be adopted and deployed in industry. Important barriers to adopting these tools include their unfamiliarity with users, their unpolished user interfaces, their poor interoperability with existing development tools and practices, and their limited support for the complex work products required by industrial software development.

Approach

Office suites, by contrast, are capable, mature, flexible, extensible, and familiar to many developers. For example, common office suites are used daily to browse Web content, produce multimedia documents, prepare presentations, and maintain budgets. These suites and other middleware-based environments can be extended and leveraged to provide familiar support for software engineering tasks. Developing and deploying innovative research tools and ideas as extensions to modern, commonly used desktop environments may ease the barriers to adoption. Users will more likely adopt tools that work in an environment they use daily and know intimately. That is, tool adoption will be improved if we specifically address the issues of cognitive support and interoperability.

The **cognitive support** of Software Engineering tools can be improved by exploiting the deep familiarity and expertise that users already have with their favorite applications and environments. We believe that building software engineering tools on top of these platforms will address the issue of cognitive support effectively.

Also, the **interoperability** of these tools can be improved significantly by leveraging recently developed middleware technologies. By exploiting technologies, such as plug-in or model-driven architectures and data exchange standards, we can address the issue of interoperability. Recently, tool builders and standards bodies have invented effective standards and interfaces for tool extension and customization.

These advances have opened new research avenues on how innovations in software engineering tools can be made more easily adopted by inserting them as extensions to commonly used office suites (e.g., Microsoft Office XP, Lotus SmartSuite, Sun StarOffice, and Corel WordPerfect Office) and middleware platforms (e.g., XML standards, SVG, scripting languages, model-driven architectures, and plug-in platforms). Our project aims to explore these avenues. Our main hypothesis is that in order for new tools to be adopted successfully, they must be compatible with both existing users and other tools. To validate this hypothesis, we will build prototype software engineering tools using open standards, popular office suites, and common middleware technology. Using these, we will conduct industrial case studies and structured tool experiments. The experience gained will be beneficial for both academic research and industrial practice.

Benefits

Developing effective techniques and strategies to overcome the software engineering tool adoption problem will have great value to the software and information technology sectors. Injecting more of the great software engineering research results into industrial practice has potentially a significant impact on the production of quality software. Thus, this research addresses two diverse markets: the software developers, who need to understand and document existing software systems, but also the researchers, who want to inject and validate their research tools in industrial development processes.

ACRE V1.0

ACRE V1.0 consists of several software visualization engines on top of various office products, including Lotus Notes, Excel, PowerPoint, and Visio. The software engineering tools in this ACRE environment interoperate using the ACRE persistence engine and SVG (Scaleable Vector Graphics). The ACRE persistence engine is implemented using IBM Websphere software platform [<http://www-3.ibm.com/software/info1/websphere/>], the OMG's Model Driven Architecture (MDA) [<http://www.omg.org/mda/>], and OTI's universal tool platform Eclipse [<http://www.eclipse.org/>]. SVG is a W3C XML standard and an effective solution for smart cross-platform graphics.



Towards Live Documents for Software Engineering Documentation

Anke Weber, University of Victoria, Canada [anke@cs.uvic.ca]

An ACRE Live Document is

- a static document with a state
- data-driven and interactive
- adapts automatically and intelligently to its context
- is implemented on top of standard office platforms

Live documents can be used to

- synchronize code and documentation automatically (e.g., keep diagrams in line with the source code)
- produce multiple output versions from one source consistently (e.g., for print, online and audio use)
- address different audience needs (e.g., user manuals for novice and expert users)
- explore the system without leaving the document
- support group collaboration

Meeting reverse engineering requirements with MS Excel, PowerPoint and Visio

RE requirement	Excel	PowerPoint	Visio
Visualize program information artefacts and architecture	Drawing tools	Custom presentations and animation	Templates for diagrams, e.g. UML and Web sites
Statistical data analysis and metrics	Built-in statistical functions and charts		
Re-document system	Report builder Synchronize with data sources	Custom presentations and animation	Synchronize with data sources
Repository based	ODBC support Excel sheets as data sources,		
Collaboration features	Protect and share workbook Track changes	Track changes and merge documents	
	Meeting scheduling and sending documents via email		
Robust	Standard functionalities to build upon; e.g. "undo"		
Data-driven and net-centric	Web services, Smart Tags, and Dashboards, Web publishing Integrate with databases (e.g., Access) and MS server environments		
Interoperability	Active X/OLE support Support for XML and SVG objects		
End-user programmable and Office automation	Macro recording, playing, and editing, Scripting with VB script (Com) Add-Ins with VBA, Dynamic Libraries with .Net		
Leverage cognitive support	Large user base: according to Microsoft more than 250,000,000 users for Office		

Towards a Live Manual

- Keeping documentation up-to-date
 - synchronize graph analysis and visualization using Excel built-in statistical functions and charts
 - synchronize visualization of statistical data using Excel charts
- Enhancing documentation capabilities of the Rigi reverse engineering system
 - different views on the graph data and statistics in Excel within one workbook
 - copy Excel views to PowerPoint for advanced presentations features
- Keeping source code and documentation in sync
 - capture Rigi graphs and display in Excel, PowerPoint and Visio
 - edit Rigi graphs in Excel, PowerPoint and Visio



ACRENotes—Leveraging Cognitive Support in Lotus Notes to build ACRE tools

Jun Ma, University of Victoria, Canada [majun@csc.uvic.ca]

What is ACRENotes?

ACRENotes is an adoption-centric reverse engineering (ACRE) tool that is developed with the popular groupware product Lotus Notes. It has the following components:

- ❑ Data Repository
- ❑ Data Visualization
- ❑ End-User Programming
- ❑ Team Cooperation

Why Lotus Notes?

- ❑ Leverage GUI and metaphors familiar to users

As a successful groupware product, Lotus Notes has been used by millions of users all over the world. Consequently, Lotus Notes users have become familiar with its GUI, metaphors, and operation processes. Building on Lotus Notes, ACRENotes can save users a lot of time and effort compared to learning a stand-alone software engineering tool.

- ❑ Utilize built-in document database to efficiently manage data

Both documents and data in Lotus Notes are managed in its own document database. Documents can be easily selected, browsed, filtered, and categorized. Furthermore, they can be handled by predefined Agents and Actions. ACRE leverages Lotus Notes' database: By converting reverse engineering data (e.g., from GXL) into documents, the users' capability to directly manipulate reverse engineering data can be maximized. Moreover, all kinds of artifacts generated by the development team can be managed easily in this database.

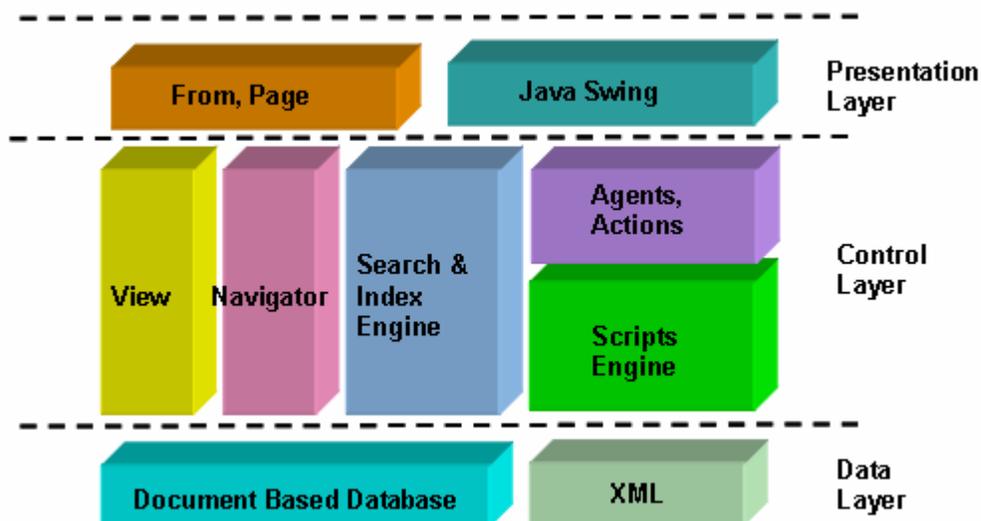
- ❑ Integrate CSCW support in reverse engineering tools

Lotus Notes' groupware features facilitate cooperation between team members. They can easily exchange project information in the form of Lotus Notes documents. Project managers can access information to coordinate and direct the team's effort. Therefore, people can work on a large reverse engineering project effectively and efficiently.

- ❑ Enable end-user programming

Users can access and manipulate their data with the help of scripting languages embedded in Lotus Notes (e.g., JavaScript). Therefore, tedious and complex tasks can be efficiently automated.

The ACRENotes Architecture





The ACRE SVG Visualization Engine (ASVE)

Jon Pipitone, University of Toronto, Canada [g1u@cdf.toronto.edu]

Holger Kienle, University of Victoria, Canada [kienle@cs.uvic.ca]

What is ASVE?

- ❑ A graph visualization engine for exploring and annotating software artifacts
- ❑ Built exclusively with SVG and ECMAScript
- ❑ Embeddable into "host" applications such as Web browsers and office tools (e.g. PowerPoint, Excel, Word)
- ❑ A live document component
- ❑ A client for the ACRE Persistence Engine

Why SVG?

Scalable Vector Graphics (SVG) is an XML grammar for Web graphics from the W3C. Advantages of SVG over other image formats are:

- ❑ infinite zooming with crisp edges (i.e., no "jaggies")
- ❑ embeddable in office applications, web pages, and XML documents
- ❑ (open source) viewers for common platforms
- ❑ XML-based format that is non-proprietary, human-readable, and searchable
- ❑ wide selection of tools available to edit and transform (XSLT, XQuery, etc.)
- ❑ small file sizes for high resolution graphics
- ❑ easy manipulation through standard APIs, such as the DOM API

What makes ASVE a good ACRE tool?

- ❑ Embeddable

Since ASVE is built using pure SVG it can easily be embedded into project documentation: PowerPoint for presentations, HTML pages for on-line browsing, Word documents for hard-copy documentation, and XML documents for further post-processing.

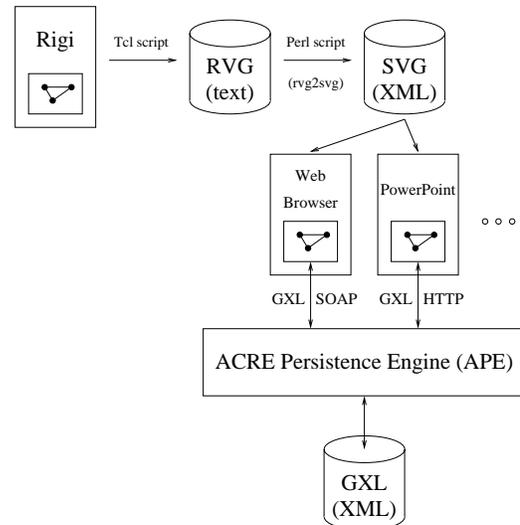
- ❑ Live and Interoperable

ASVE can act as a lightweight, platform independent client of the ACRE Persistence Engine. Regardless of the "hosting" application, ASVE can always connect to its data in the ACSE repository.

- ❑ Rich Knowledge Store

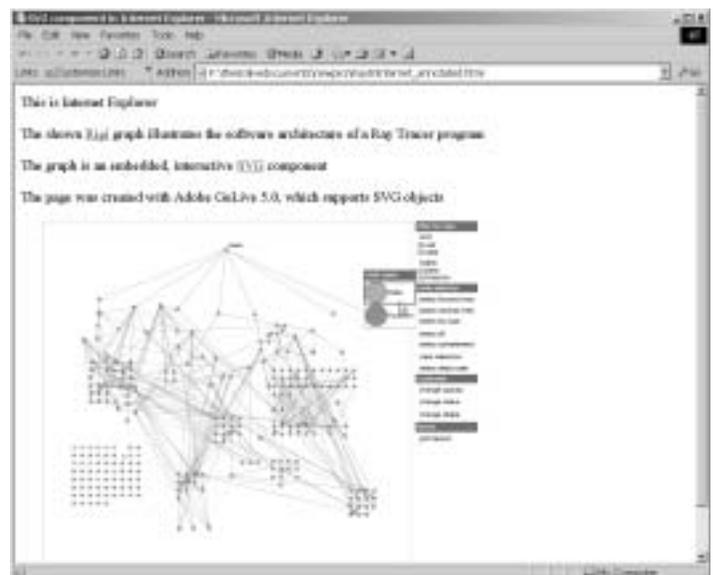
With ASVE, a single file encapsulates a snapshot of a Rigi reverse engineering view along with all the graph manipulation functionality needed for further exploration (see "ASVE Architecture").

ASVE Architecture



ASVE in Internet Explorer

The illustration below shows an extracted Rigi view—a call-graph of a ray-tracer application—in ASVE. Here, ASVE is "hosted" by Internet Explorer. The user can filter, rearrange, layout, annotate, display attributes, and change the visual characteristics of nodes and arcs in order to better understand the architecture of the software system.





The ACRE Persistence Engine

Will Kastelic, University of Victoria, Canada [will@cs.uvic.ca]

David Zwiers, University of Victoria, Canada [dzweirs@csc.uvic.ca]

What is it?

- ❑ A feature-rich, and extensible middleware system upon which to develop software engineering tools.
- ❑ Software engineering tool developers can build their custom applications on the Engine.
- ❑ Build on what is already in place – re-use!

An Example – Software Requirements Engineering

Some of the features that may be found in a Requirements Toolkit include:

- ❑ Data Repository processing
 - Requirements docs
 - Import/export docs
 - Dictionaries, glossaries
- ❑ Views
 - Graphic output
 - User-defined
- ❑ Requirements processing
 - Support standards
 - Requirements tracing
 - Sort/analyze metrics
- ❑ Integration
 - with popular IDEs
 - with popular SE tools
- ❑ Version control/change mgmt
- ❑ Secure access control
- ❑ Extras
 - Whiteboarding
 - On-line discussion
 - Custom calculations

Before the ACRE Persistence Engine

Developers have to write code for all aspects of the tool—from the basic infrastructure, such as client interfaces, the repository access, to the code which makes the tool unique. The lack of a standards-based design leads to more one-of-a-kind implementations.

With the the ACRE Persistence Engine

Tool developers can focus on creating custom code for new required features, as the Engine provides the basic components. Designers and developers can focus on creating code in support of the cognitive principles behind ACRE.

Architecture of the ACRE Persistence Engine

The ACRE Engine can be extended to provide the functionality needed in the Software Engineering tool/application you are developing.

The architecture utilizes

- ❑ International standards (e.g., MDA, WSDL, SOAP, ODBC)
- ❑ Common data format (SVG/GXL/XML) for 3rd party integration
- ❑ IBM's WebSphere Application Server

The architecture provides

- ❑ Multiple perspectives/views
- ❑ Data repository control
- ❑ Version control
- ❑ Web and e-mail services;
- ❑ Web authoring toolkit
- ❑ XML data interchange
- ❑ Java interface
- ❑ Debugging and profiling
- ❑ End-user programmability
- ❑ API for multi-client applications
- ❑ Scripting

