

Understanding Object Oriented Software

Melissa P. Chase

781-271-2113 • pc@mitre.org

MITRE Sponsored Research

The logo for the MITRE Technology Program, featuring a stylized graphic of stacked blocks in yellow, orange, and blue to the left of the text.

MITRE
Technology
Program

Problem

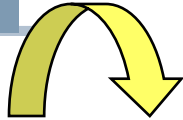
- **Developing and sustaining large, reliable software systems is a challenge.**
- **Understanding software assets is necessary for decreasing the cost and improving the quality of software by supporting**
 - **Iterative development, training**
 - **Reuse of legacy components**
 - **Assessment of software quality, performance, security, and other features.**

Background

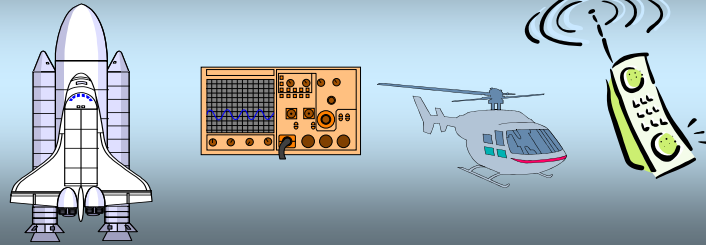
Software
Developers



Using
S/W architecture
Design Patterns
Programming Idioms



Large, Complex Systems



Source Code (today's systems using
object-oriented software - C++, Java)

Analysts, Funding
Agencies
??????



What have we got here?

How and where can we safely
and effectively use it?

If we know the structures used, can we answer these questions?



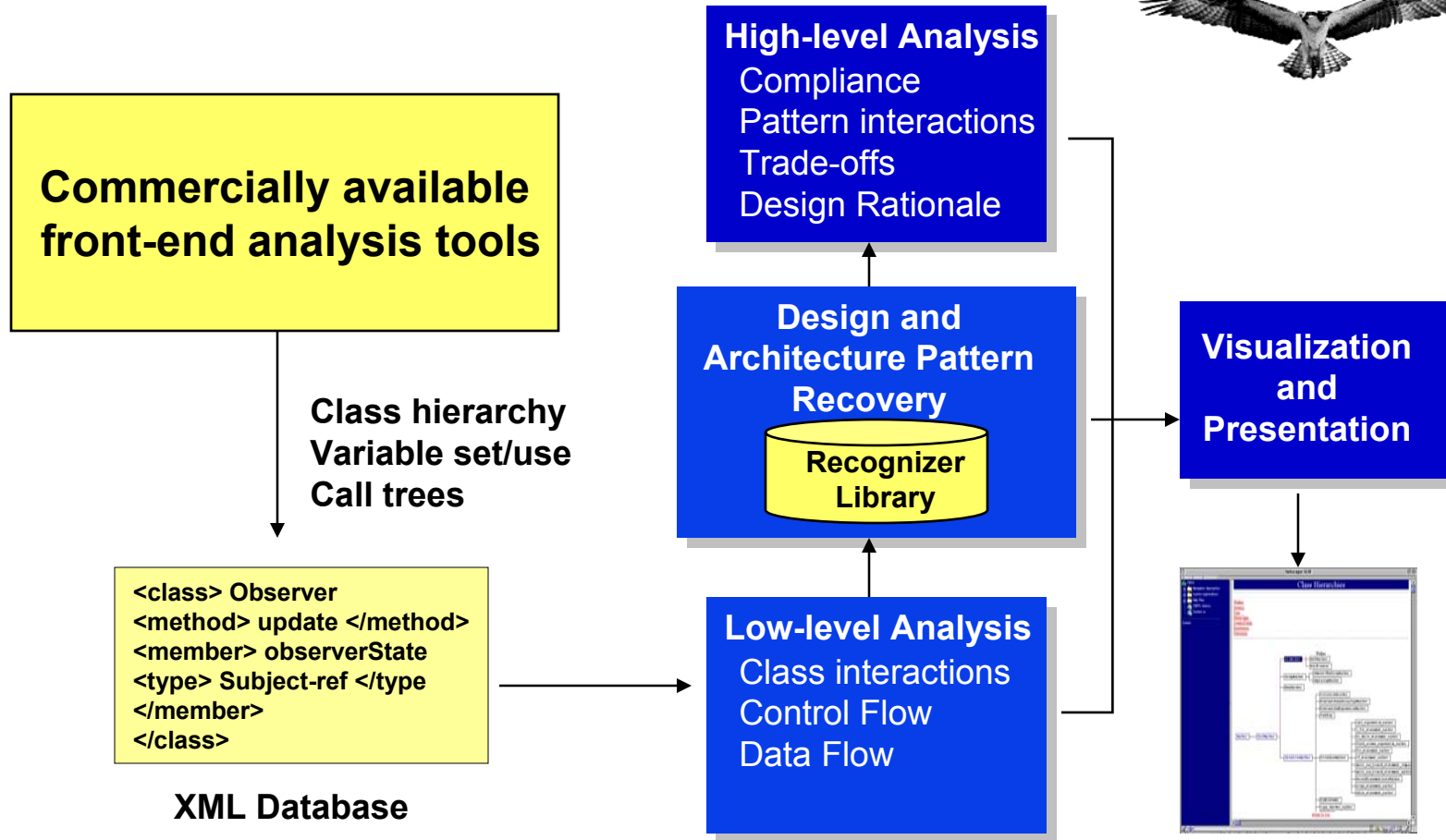
Objective

- **Develop the capability to understand delivered software applications by**
 - **Recognizing the use of “design patterns” -- best practices in software construction**
 - **Identifying the design rationale that led developers to use specific design patterns**
 - **Profiling software qualities inherent with the use of these patterns**

Activities

- **Build Osprey, a small-footprint analysis tool containing**
 - **Class/method interaction extractors for C, C++, Java - using Imagix and Together Integrated Development Environments**
 - **Extensible specification language for design pattern recognizers. 89 recognizers for 62 pattern types**
 - **Pattern-centric Web-based documentation**
 - **Truth-maintenance inferences involving pattern applicability and consequences (with respect to software qualities)**

Osprey Architecture



Demonstration



Osprey-Generated Documentation Packages

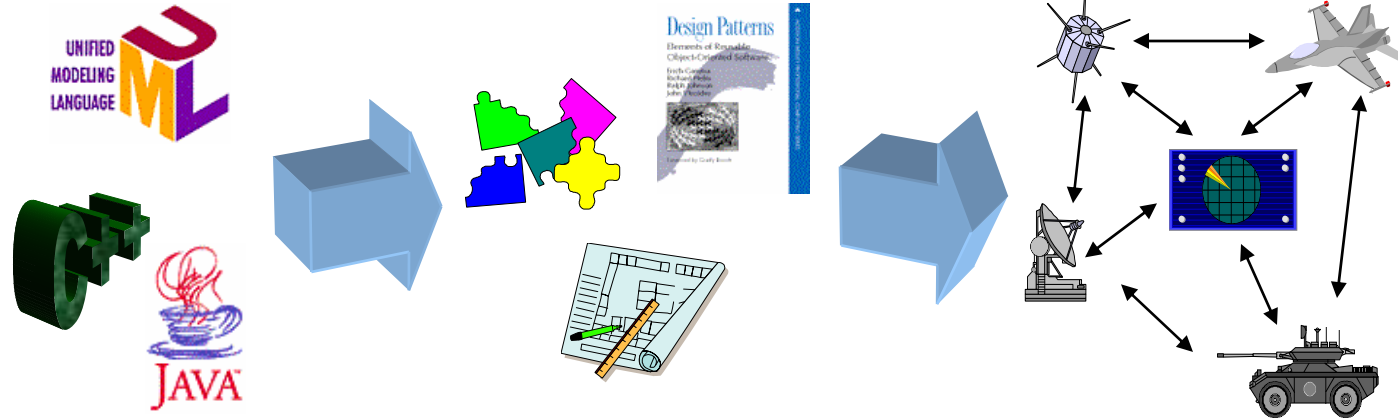
The image displays four overlapping browser windows showing generated documentation for ACE (Attack Capability Engine). The windows are:

- Pattern Instances for ACE:** A table listing various patterns with columns for Name, Instance, Pattern, Architecture, Structure, and Creator.
- Ordered classes for ACE:** A list of class names, such as ACE_Base, ACE_Base_Ext, ACE_Base_Ext2, ACE_Base_Ext3, ACE_Base_Ext4, ACE_Base_Ext5, ACE_Base_Ext6, ACE_Base_Ext7, ACE_Base_Ext8, ACE_Base_Ext9, ACE_Base_Ext10, ACE_Base_Ext11, ACE_Base_Ext12, ACE_Base_Ext13, ACE_Base_Ext14, ACE_Base_Ext15, ACE_Base_Ext16, ACE_Base_Ext17, ACE_Base_Ext18, ACE_Base_Ext19, ACE_Base_Ext20, ACE_Base_Ext21, ACE_Base_Ext22, ACE_Base_Ext23, ACE_Base_Ext24, ACE_Base_Ext25, ACE_Base_Ext26, ACE_Base_Ext27, ACE_Base_Ext28, ACE_Base_Ext29, ACE_Base_Ext30, ACE_Base_Ext31, ACE_Base_Ext32, ACE_Base_Ext33, ACE_Base_Ext34, ACE_Base_Ext35, ACE_Base_Ext36, ACE_Base_Ext37, ACE_Base_Ext38, ACE_Base_Ext39, ACE_Base_Ext40, ACE_Base_Ext41, ACE_Base_Ext42, ACE_Base_Ext43, ACE_Base_Ext44, ACE_Base_Ext45, ACE_Base_Ext46, ACE_Base_Ext47, ACE_Base_Ext48, ACE_Base_Ext49, ACE_Base_Ext50, ACE_Base_Ext51, ACE_Base_Ext52, ACE_Base_Ext53, ACE_Base_Ext54, ACE_Base_Ext55, ACE_Base_Ext56, ACE_Base_Ext57, ACE_Base_Ext58, ACE_Base_Ext59, ACE_Base_Ext60, ACE_Base_Ext61, ACE_Base_Ext62, ACE_Base_Ext63, ACE_Base_Ext64, ACE_Base_Ext65, ACE_Base_Ext66, ACE_Base_Ext67, ACE_Base_Ext68, ACE_Base_Ext69, ACE_Base_Ext70, ACE_Base_Ext71, ACE_Base_Ext72, ACE_Base_Ext73, ACE_Base_Ext74, ACE_Base_Ext75, ACE_Base_Ext76, ACE_Base_Ext77, ACE_Base_Ext78, ACE_Base_Ext79, ACE_Base_Ext80, ACE_Base_Ext81, ACE_Base_Ext82, ACE_Base_Ext83, ACE_Base_Ext84, ACE_Base_Ext85, ACE_Base_Ext86, ACE_Base_Ext87, ACE_Base_Ext88, ACE_Base_Ext89, ACE_Base_Ext90, ACE_Base_Ext91, ACE_Base_Ext92, ACE_Base_Ext93, ACE_Base_Ext94, ACE_Base_Ext95, ACE_Base_Ext96, ACE_Base_Ext97, ACE_Base_Ext98, ACE_Base_Ext99, ACE_Base_Ext100.
- Analysis by Directories and Files for ACE:** A legend for the analysis grid, showing categories like 'box colors', 'facts', 'egg file', 'shdbox colors', 'Creates and Behaviors', 'Creates and Structuring', 'Structuring and Behaviors', 'Behaviors', 'Creates', 'Structuring', and 'Architecture'.
- Pattern Instance Detail:** A detailed view of a specific pattern instance, showing its name, instance, pattern, architecture, structure, and creator.

Impacts

- **Support for MITRE's sponsors**
 - **quality assessment for IBS**
 - **OO software understanding for SIAP**
- **Analysis of MITRE-developed software:**
 - **LEdit, MSIM modeling and simulation tools**
- **Developing software security recognition for C++ and Java code**
- **Publishing at reverse engineering conference**

Future Plans



Using
Source Code
Models
Documentation

Reverse Engineer
Patterns
Components
Architectures



Map design elements to
functionality
Analyze design trade-offs
Ensure architectural
conformance
Predict operational
success