OpenOME distilled
@ Early Requirements Seminar, 2005

Presented by Yijun Yu

A Tool for Goal/Agent/Aspect-oriented Requirements Engineering
http://sourceforge.net/projects/openome
http://www.cs.toronto.edu/~yijun/OpenOME.html
Abstract

- OpenOME is a goal/agent/aspect-oriented requirements engineering tool. In this talk, we explain the current development of the OpenOME, as a result of reengineering the legacy OME tool. We explain the unique features of OpenOME and the improvements on usability, extensibility and interoperability. Currently OpenOME supports some advanced research topics such as goal modeling and analysis, ontology queries, requirements knowledge reuse. In the near future, it will support Web-service based editing, weaving of requirements goal aspects, discovery and application of requirements patterns, viewpoint extraction and applications through ontology queries, etc. OpenOME is 100% open-source and all contributions are welcome. At the end, we show how you can contribute.
Agenda

1. Motivations
2. What have been done since we started?
3. Some research topics
4. A tool of you, by you and for you: Any suggestions, contributions are extremely welcome!
5. How can you contribute?
1. Motivations: Reengineer OME

- **OME** = *Organizational Modeling Environment.*
  [http://www.cs.toronto.edu/km/ome](http://www.cs.toronto.edu/km/ome)

  It was part of the *Tropos* project to support goal-oriented and agent-oriented requirements engineering methodologies (>5 years of development, >10 man-year efforts)

- OME has been used by more than 130 users across the globe

- Every OME user must sign an agreement with *Techne* because the core *Telos* knowledge base is a binary module protected by the license

- To enlarge the user-base and make it fully-extensible, we decide to open-source it last year ... **OpenOME**
  - Replace the *Telos* DLL module with an open-source module
  - Reengineer the code base to pure Java
  - Reshape the plug-in architecture
  - Integrate with other modules

- Some quality improvements
  - Usability, Scalability, Extensibility, Reusability, etc.
What is OpenOME good for?

• OpenOME is a general graph editor, supporting conceptual modeling for Entity Relationships (ER), goal models (NFR, i*, GRL), etc.

• It is designed to support requirements engineering
  – Goal-oriented: goal reasoning through label propagation (GRL, i* strategic rationale, NFR)
  – Agent-oriented: group goals into agents (i* strategic dependency)
  – Aspect-oriented: group NFR in the “v”-graphs into aspects

• It can interoperate with other graph editors
  – ConceptBase (OTelos) ......................... already in OME
  – Stanford Protégé..............Ontology Editors..... (OWL)
  – AT&T Graphviz............Graph Layouts........ (DOT)
  – Microsoft Visio.............Scalability.............(XSLT)
  – Rational unified process....Model-driven...... (EMF/XMI)
Acknowledgement

OpenOME is a forged effort

Progress until now …

• Eclipse development
• Copy/Paste support (with Zhifeng Liu)
• Interact with Protégé
  (Cascon’04 demo, with Eric, Jennifer, Frank, Jane, …)
• Load Visio model (with Jorge and Marcel)
• Layout through graphviz (with Xiaoxue)
• Load/Save Telos Knowledge Base without jtelos.dll
  (with Xiaoxue)
• Reengineering to the MVC pattern
• Q7 representations (with Julio)
How to begin with …

• Download the recent SDK at
  – http://sourceforge.net/projects/openome
  – http://www.cs.toronto.edu/~yijun/OpenOME.html

• Follow the instructions at

• You can start OpenOME in various ways:
  – Command line: “run” versus “run_protege”
  – “Run” menu inside Eclipse
  – Windows: Associate file extensions “.tel”, “.q7”, “wdx” to “run.bat” under the OpenOME project directory
Input and Output Formats

• run.bat projects/telos/q7/streamline.q7
• An input file can be in one of the following formats:
  – Telos
  – XML (E.g. Visio) provided there is an XSLT to convert it into Telos
  – Any programming language as long as you can create a parser using JavaCC
    (E.g., Telos, Q7)
  – DOT layout file, later …
  – Java/EMF/XMI model, later …
• An output file of OpenOME can be any of the following:
  – Telos, the de facto format
  – OTelos (SML), to communicating with ConceptBase
  – Any output format supported through Protégé (E.g., OWL)
  – PNG, the model can be saved into (File/Extract …) a high-quality picture
  – DOT, the graph description format for Graphviz is saved when you do “File/Layout…”
2. What we have done?

1. Eclipse development
2. Copy/Paste support (with Zhifeng Liu)
3. Interact with Protégé
4. Load Visio model (with Jorge and Marcel)
5. Layout through graphviz (with Xiaoxue)
6. Load/Save Telos Knowledge Base without jtelos.dll (with Xiaoxue)
7. Q7 representations (with Julio)
2.1 Eclipse development

Run Menus

Debugging Menu

Double Click Here
2.2 Copy and paste

The paste operation is not blind
2.3 Save to Protégé

Single Click here!
2.4 Visio

• i* in Visio

• The Visio graph loaded into OpenOME
2.5 Auto layout

Before

After
2.6 Platform independence (MJF)
2.7 Q7 support
What is Q7?

Q7 = 5W2H, an abstract language to classify NFR/i* knowledge for reuse

- When context-oriented
- Who agent-oriented
- Why goal-oriented
- What object-oriented
- Where aspect-oriented
- How system/function-oriented
- How much metric-oriented
3. Some research topics

1. Q7, supporting the quality-based software reuse using aspect-oriented technology
   It also supports checklist requested by the KHP group

2. Ontology and Viewpoints:
   find errors
   actor class view
   Future work: PAL (Protégé Axiom Language) Queries
   Future work: Front-end for Homomorphism Views Merging

3. Reasoning with Goal Models through label propagation
   Future work
   No time to expand the following topics:

4. Requirements and other SE phases
   1. Forward Engineering:
      From Stakeholder Goals to High-variability Software Designs
   2. Reverse Engineering:
      Refactoring Source Code into Goal Models
   3. Conceptual modeling: EclipseUML Ontology Telos

5. OmniGraphEditor + OpenOME: Web-service oriented implementation of
   the group editing of the requirements goal models (course project of
   ECE450)
3.1 Q7 language for aspect reuse

```cpp
<car>::design { &
design[wheels]
design[shape] { &
  (true)=>design[head] => ++widening
  (true)=>design[body] => ++constant
  (true)=>design[tail] => ++narrowing
} => ++streamline
design[engine] => ++powerful
}
speed { &
  streamline
  powerful
}
streamline{ &
  widening
  constant
  narrowing
} => ++beauty
```
Q7 loaded in OpenOME

- Initially, the agents are compacted
- You can expand them into the Strategic Rationale view

Single Click here!
Result i* SR model
A pattern to create strategic dependencies (not done yet)
Where are the aspects?

```
<car>::design { &
  design[wheels]
  design[shape] { &
    (true)=>design[head] => ++widening
    (true)=>design[body] => ++constant
    (true)=>design[tail] => ++narrowing
  } => ++streamline
  design[engine] => ++powerful
}

speed { &
  streamline
  powerful
}

streamline{ &
  widening
  constant
  narrowing
} => ++ beauty
```

```
<car>::design { &
  design[wheels]
  design[shape] { &
    (true)=> design[head]
    (true)=> design[body]
    (true)=> design[tail]
  }
  design[engine]
}

<speed>::speed { &
  streamline<++*[shape]
  powerful<++*[engine]
}

<beauty>::beauty {&
  streamline<++*[shape]
}

streamline { &
  widening <=++*[head]
  constant <=++*[body]
  narrowing<=++*[tail]
}
```
Automatically woven
Example 1. Media Shop
Example 2. MeetingScheduler
Example 3. Middleware: CORBA
Q7 is a key to reuse NFR
3.2 Ontology saved (MJF)
Construct an Ontology query
Results 1 Errors highlighted
Create another query
Results 2: Actor class view
Work with other visual plugins (TGViz, Jambalava)
3.3 Quantitative label propagation

• Telos parser to convert the NFR goal model into an input file for the goal reasoning tool developed in Trento

• The result can be saved into DOT and visualized in GraphViz

• Example, Media shop
Media Shop example
Result of the label propagation
3.4 Not just requirements
3.4.1 Generating High-variability designs from Goal Models

Annotate the goal model with light-weight design information to derive the following high-variability views

• Feature models
• Statecharts
• Component-connector models
• Aspects
• Web services (BPEL)
Feature model generation

(A) g1
   AND
   AND
   g2  g3

(B) g1
   OR
   OR
   g2
   NOP
   g3

(C) g1
   OR
   OR
   g2
   g3

(D) g1
   OR
   OR
   g2  g3

f1

f2  f3
mandatory

f1
optional

f1
alternative

f1
or

f2  f3
Example
Statecharts generation

(a) Goal: name
FormalDef:
P => \Diamond Q

(b) Goal: name
FormalDef:
P => \Box Q

---

(1) (2) (1) (2)
Example

(a)

(b)
Component-connector view

\[
\begin{align*}
\text{g} & \quad \text{I: } i_1, i_2 \\
& \quad \text{O: } o_1, o_2 \\
\text{g1} & \quad \text{I: } i_1, i_2 \\
& \quad \text{O: } o_1, o_2 \\
\text{g2} & \quad \text{I: } i_1, i_2 \\
& \quad \text{O: } o_1, o_2 \\
\end{align*}
\]

interface type I0{
\[
\begin{align*}
G(\text{IN } i_1, \text{ IN } i_2, \text{ OUT } o_1, \text{ OUT } o_2); \\
\end{align*}
\]
}

Interface type I1 {
\[
\begin{align*}
G_1(\text{IN } i_1, i_2, \text{ OUT } o_1, \text{ OUT } o_2); \\
\end{align*}
\]
}

Interface type I2 {
\[
\begin{align*}
G_2(\text{IN } i_1, i_2, \text{ OUT } o_1, \text{ OUT } o_2); \\
\end{align*}
\]
}
Example.
Generated views
3.4.2 Reverse engineering goal models from legacy code

1. Extract Methods Refactoring

- Legacy source code
- Refactored source code

- Is structured?
  - Yes
  - Refactored source code
  - Structured program
  - Eliminate GOTO
  - High-level Statechart
  - Goal Model with NFRs
  - Test to identify NFRs
  - Goal Model with NFRs
  - Analyze quality metrics to identify softgoals

- No
  - Construct Hammock graph
  - Extract States/Transitions Refactoring
  - Statecharts
  - Goal Model with Softgoals
Example. Columba Refactoring

- Search “Java email client” in Google, you will find this software
- It is open-source
- It has 140 KLOC in Java
- It also has plug-in patterns
- First thing, we modify the code base to fit Eclipse development (moving packages, i.e., move all “src” subdirectories including plug-in projects under the same “src” directory)
A screenshot
Where to look at first?

• Secondly, we look for the main routine from the manifest in the JAR file

Manifest-Version: 1.0
Ant-Version: Apache Ant 1.6.2
Created-By: 1.4.2_06-b03 (Sun Microsystems Inc.)
Main-Class: org.columba.core.main.Main
Sealed: false
Class-Path:  lib/usermanual.jar  lib/junit.jar  lib/lucene-1.3-final.jar
  lib/commons-cli-1.0.jar  lib/jwizz-0.1.2.jar  lib/plastic-1.2.0.jar  lib/jhall.jar  lib/forms-1.0.4.jar  lib/ristretto-1.0_RC2.jar  lib/jscf-0.2.jar  lib/macchiato-1.0pre1.jar  lib/rapuccino-1.0pre1.jar  lib/winpak.k.jar  lib/jniwrap-2.4.jar  lib/jdom.jar  lib/jpim.jar  lib/je.jar  ${lib.jdic}
The Main routine

public static void main(String[] args) {
    Main.getInstance().run(args);
}

Thus we look at “run” routine, which has 81 lines of code
The Run routine

```java
public void run(String args[]) {
    ColumbaLogger.createDefaultHandler();
    registerCommandLineArguments();
    // handle commandline parameters
    if (handleCoreCommandLineParameters(args)) {
        System.exit(0);
    }
    // prompt user for profile
    Profile profile = ProfileManager.getInstance().getProfile(path);
    // initialize configuration with selected profile
    new Config(profile.getLocation());
    // if user doesn't overwrite logger settings with commandline arguments
    // just initialize default logging
    ColumbaLogger.createDefaultHandler();
    ColumbaLogger.createDefaultFileHandler();
    for (int i=0; i<args.length; i++) {
        LOG.info("arg["+i+"]="+args[i]);
    }
    ...
```
The Run routine refactored

```java
public void run(String args[]) {
    ColumbaLogger.createDefaultHandler();
    registerCommandLineArguments();
    ComponentPluginHandler handler = register_plugins();
    handler.registerCommandLineArguments();
    handle_commandline_parameters(args);
    Profile profile = prompt_user_for_profile();
    initialize_configuration_with_selected_profile(profile);
    initialize_default_logging(args);
    SessionController.passToRunningSessionAndExit(args);
    enable_debugging_repaint_manager_for_swing_gui_access();
    StartUpFrame frame = show_splash_screen();
    register_protocol_handler();
    load_user_customized_language_pack();
    initialize_plugins(handler);
    load_plugins();
    set_look_and_feel();
    init_font_configurations();
    set_application_wide_font();
    hide_splash_screen(frame);
    handle_commandline_arguments_of_the_modules(handler);
    restore_frames_of_last_session();
    ensure_native_libraries_initialized();
    post_startup_of_the_modules(handler);
}
```
Identify NFR and introducing softgoals

```java
public boolean usability = false;
public boolean usability_language_customization = false;
public boolean usability_assured_progress = false;
public boolean usability_look_and_feel = false;
public boolean usability_font_configuration = false;
public boolean extensibility = false;
public boolean maintainability_debugging = false;
public boolean maintainability_logging = false;
public void run(String args) {
    if (maintainability_logging) ColumbaLogger.createDefaultHandler();
    registerCommandLineArguments();
    ComponentPluginHandler handler = register_plugins();
    handler.registerCommandLineArguments();
    if (extensibility) handle_commandline_parameters(args);
    Profile profile = prompt_user_for_profile();
    initialize_configuration_with_selected_profile(profile);
    if (maintainability_logging) initialize_default_logging(args);
    SessionController.passToRunningSessionAndExit(args);
    if (maintainability_debugging) enable_debugging_repaint_manager_for_swing_gui_access();
    StartUpFrame frame = null;
    if (usability_assured_progress) { frame = show_splash_screen(); }
    register_protocol_handler();
    if (usability_language_customization) load_user_customized_language_pack();
    initialize_plugins(handler);
    if (extensibility) load_plugins();
    if (usability_look_and_feel) set_look_and_feel();
    init_font_configurations();
    if (usability_font_configuration) set_application_wide_font();
    if (usability_assured_progress) hide_splash_screen(frame);
    if (extensibility) handle_commandline_arguments_of_the_modules(handler);
    restore_frames_of_last_session();
    if (extensibility) ensure_native_libraries_initialized();
    if (extensibility) post_startup_of_the_modules(handler);}
```
The system without the NFRs
The goal model
3.4.3 meta model implementation
3.5 Web service-based OpenOME

• OmniGraphEditor is a web service that keeps shared channels
• OpenOME communicates to the Web service, open-up cooperative requirements engineering possibilities
• Client/Server versus Peer-to-Peer
• Let’s hope the ECE450 students can deliver something at the end of the course
4. Relation to your research

1. This is a tool of you, by you and for you
2. Any suggestions, contributions are extremely welcome!
5. How can you contribute?

http://sourceforge.net/projects/openome

To contribute:
1. Apply a developer account at source forge
   http://sourceforge.net/account/newuser_emailverify.php
2. Send me email, I will add your account to the developer member list
3. Commit the source code into CVS
   CVS -d:ext:developername@cvs.sourceforge.net:/cvsroot/openome
4. Feature request
   http://sourceforge.net/tracker/?group_id=110573&atid=656743
5. Bug report
   http://sourceforge.net/tracker/?group_id=110573&atid=656742