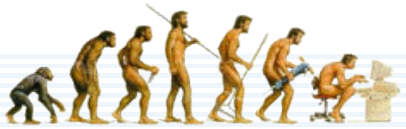
SOCNE 2008

# A Case Study on Software Evolution towards Service-Oriented Architecture

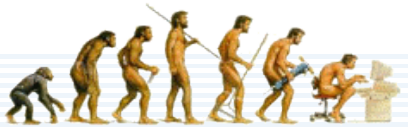
*Boni García*
*bgarcia@dit.upm.es*

*26th March 2008, Okinawa*



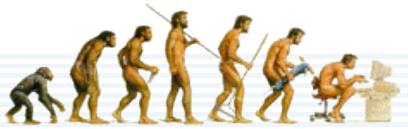
# 0. Table of contents





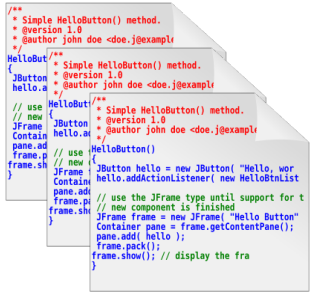
# 1. Introduction

- Maintenance costs in software: $$$
- Maintainability: capability of the software product to be modified (ISO 9126)
- SOA systems: low coupling, high maintainability, less costs
- Problem at hand: How to evolve a legacy system to SOA??

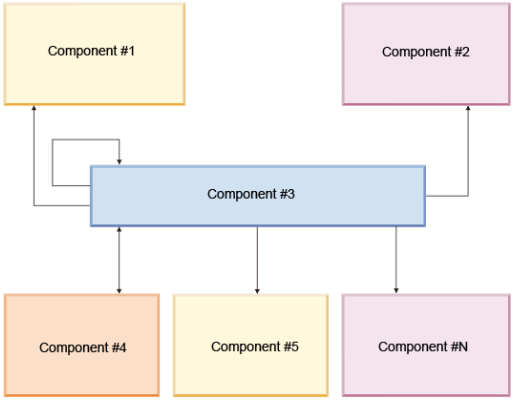


# 2. Evolution to SOA process

I. Architecture recovery



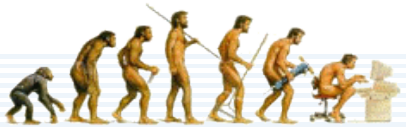
**QAR workflow**



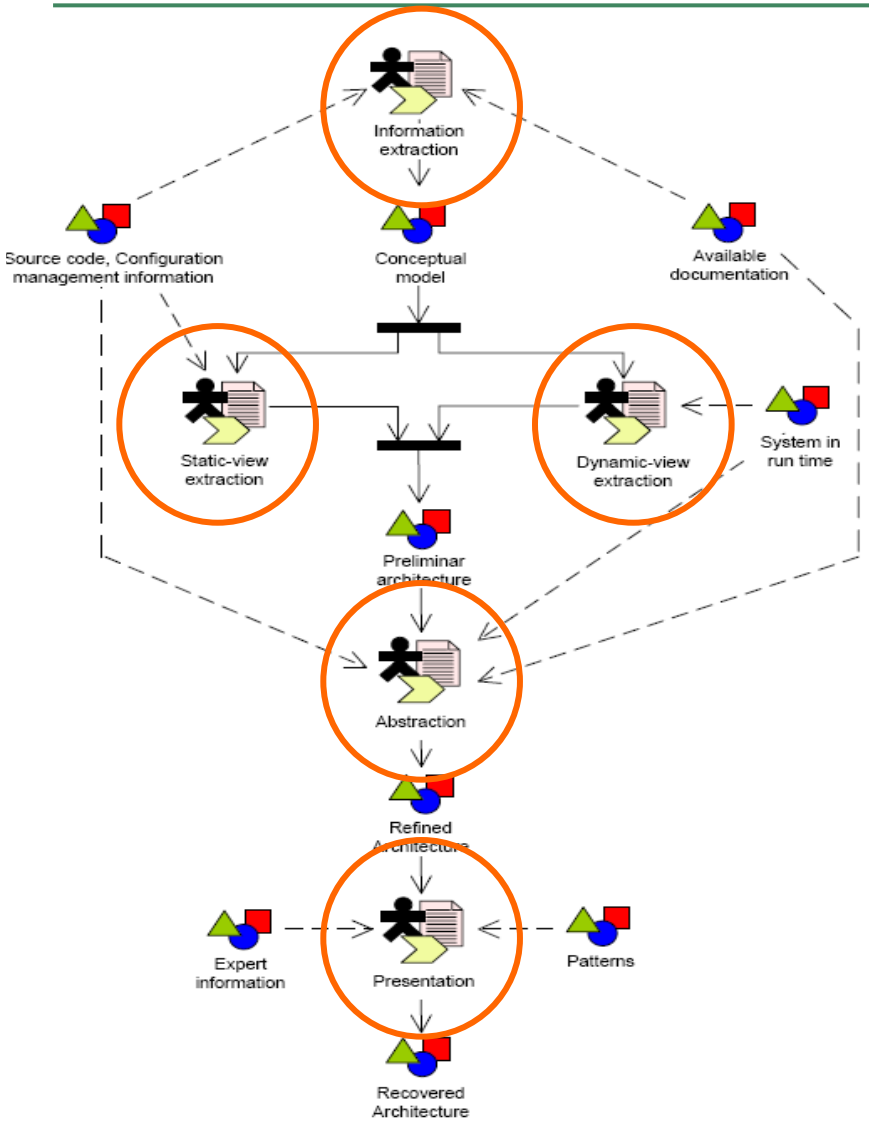

II. Evolution planning

a. Architecture selection
b. Define evolution cycles
c. Plan evolution cycles
d. Preliminary feasibility check

III. Evolution execution



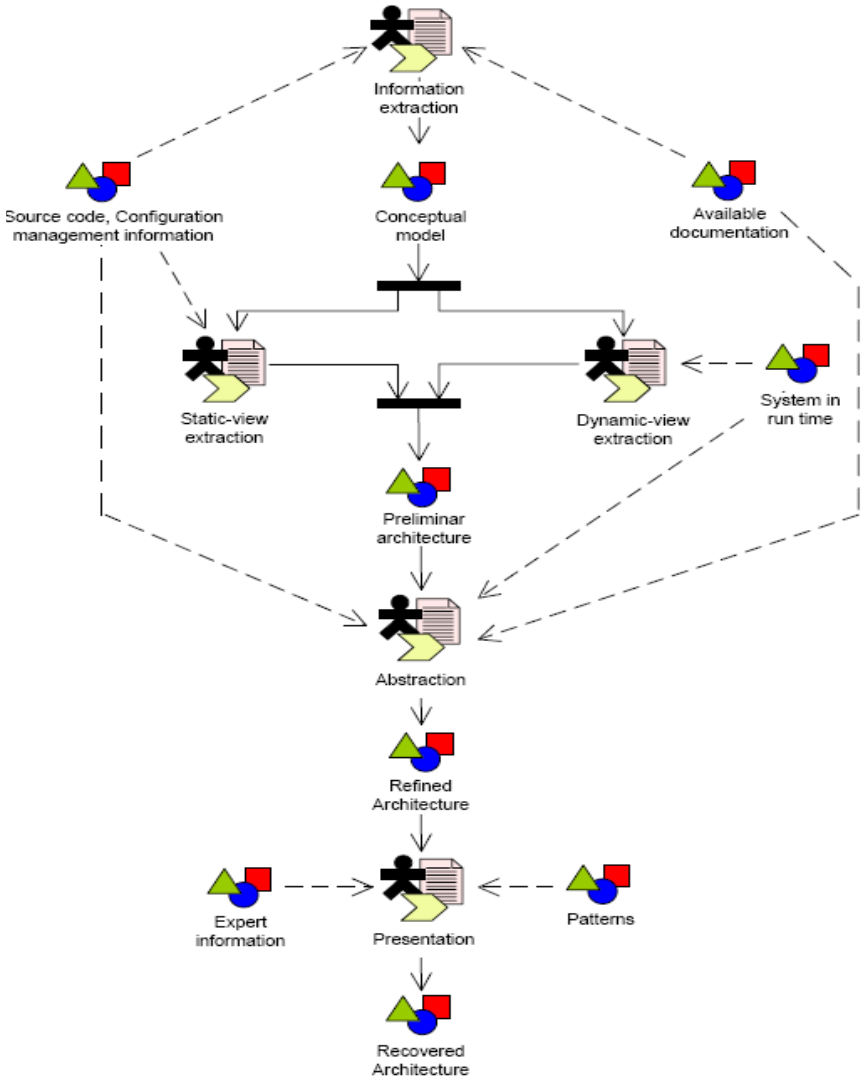
# 3. Case study: I. Architecture Recovery



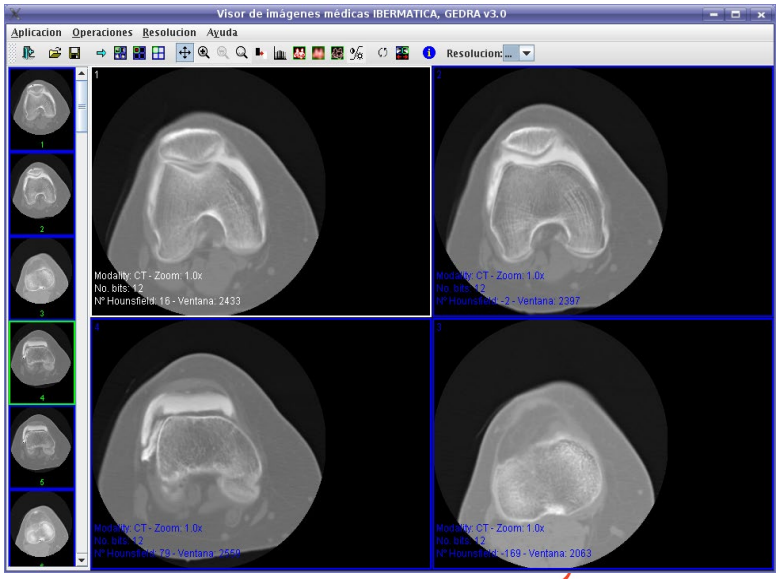

QAR Architecture Recovery

- QAR defines a generic workflow for architecture recovery
- Designed with OMG SPEM notation.
- Five processes:
  - Information extraction
  - Static-view extraction
  - Dynamic-view extraction
  - Abstraction
  - Presentation

# 3. Case study: I. Architecture Recovery

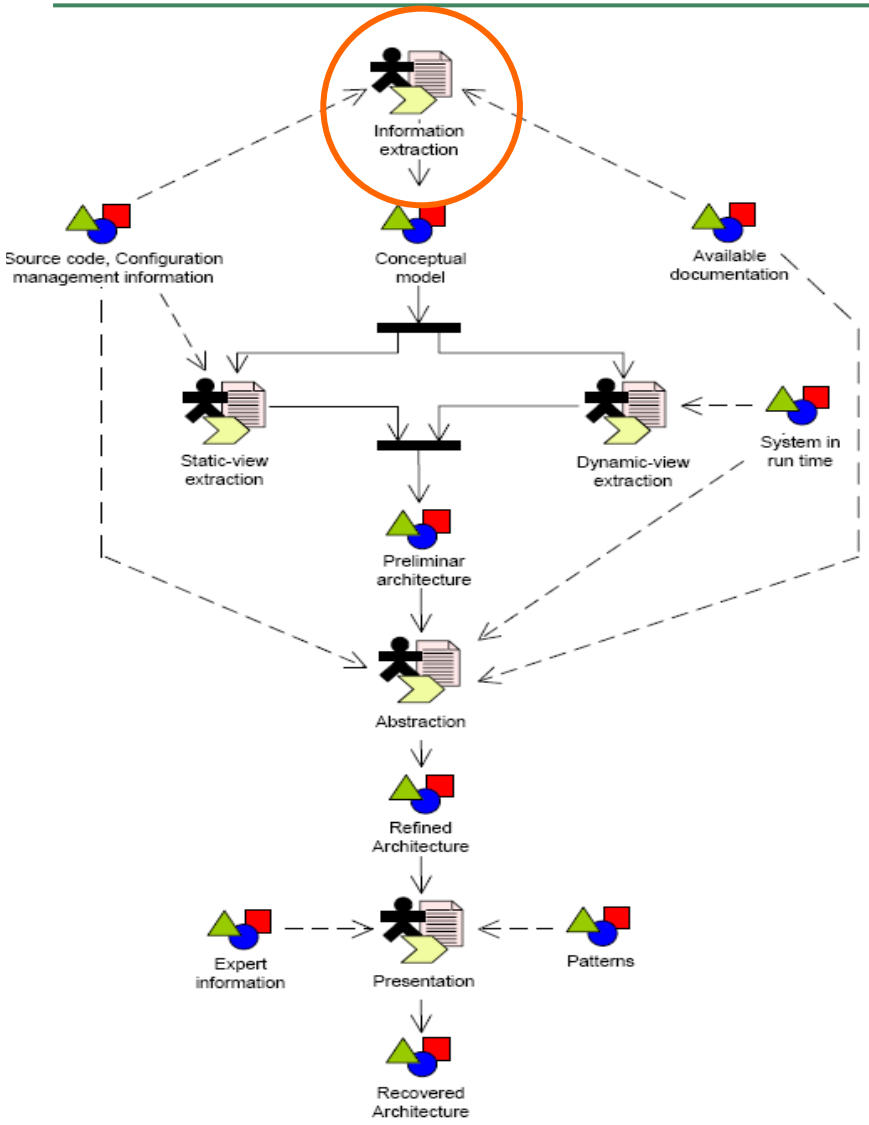



- **Legacy System to recover its architecture with QAR:**
  - **Medical Image Viewer**

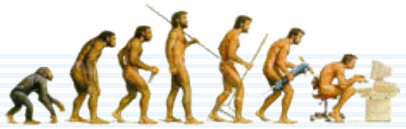



- **Developed in Java**
- **Used in several Spanish hospitals**

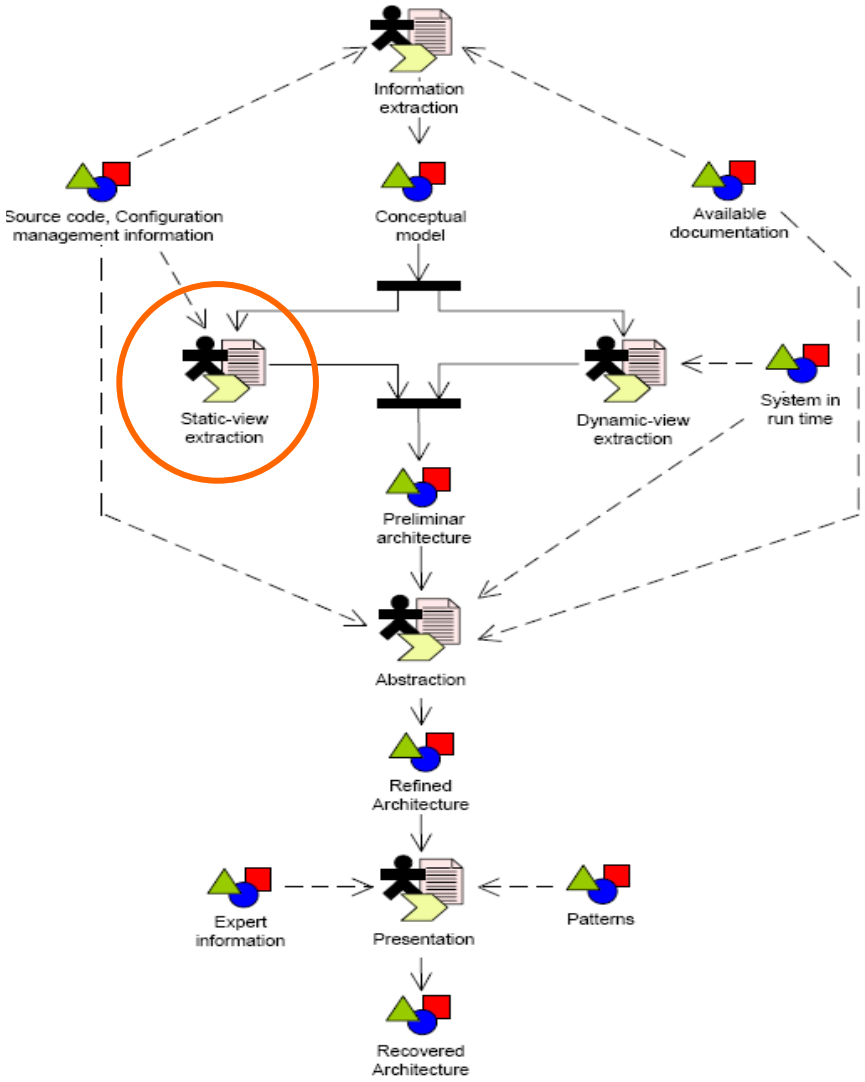
# 3. Case study: I. Architecture Recovery



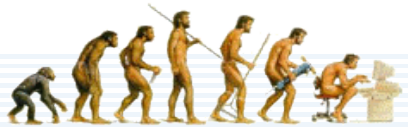

1. **Information extraction:**
   - **Source code analysis**
     - **Lines of code: 9973**
     - **Number of classes: 211**
     - **Number of packages: 17**
   - **Javadoc analysis**
     - **Image Format: BMP, JPEG, DICOM**
     - **Imaging: AWT, Java2d, JAI**
     - **GUI: Swing**
   - **User manual analysis**
     - **Functionality**



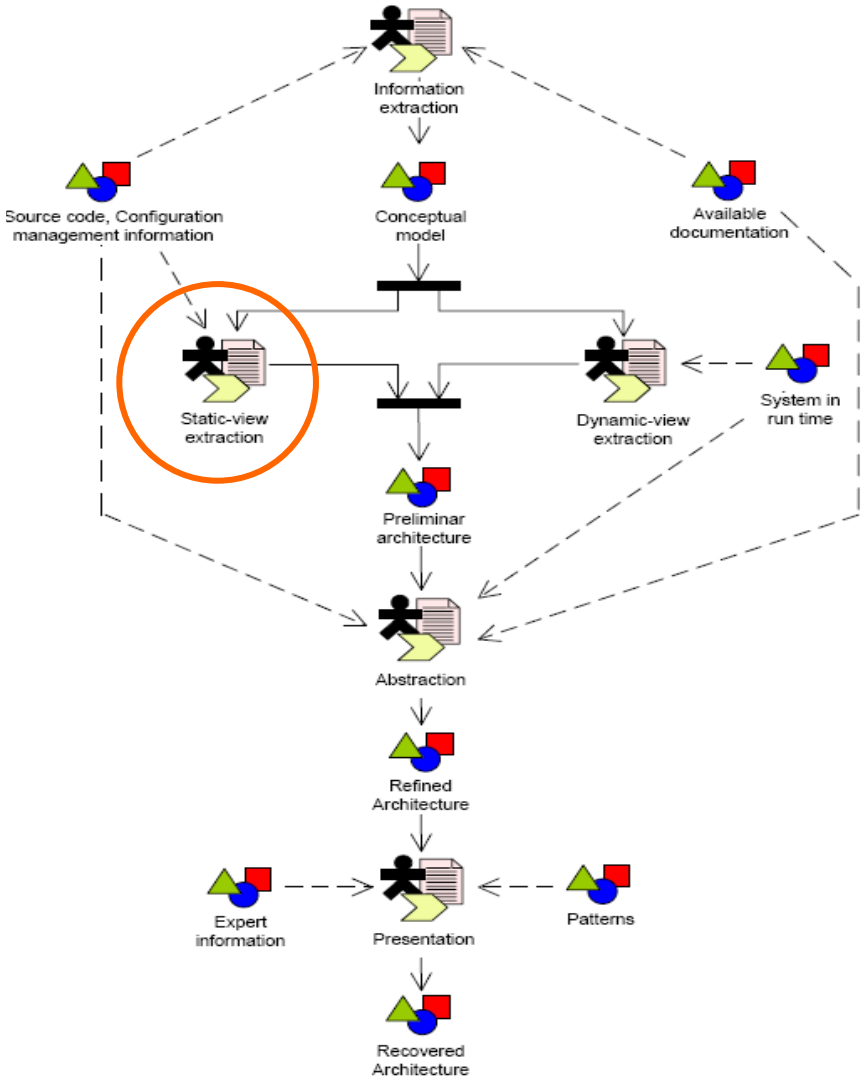
# 3. Case study: I. Architecture Recovery




2. **Static-view extraction. Tools:**
   - **Jude Community Edition**
     - **Generates UML from code**
     - **Detecting dependencies and inheritance**
     - **1st approach to architecture**

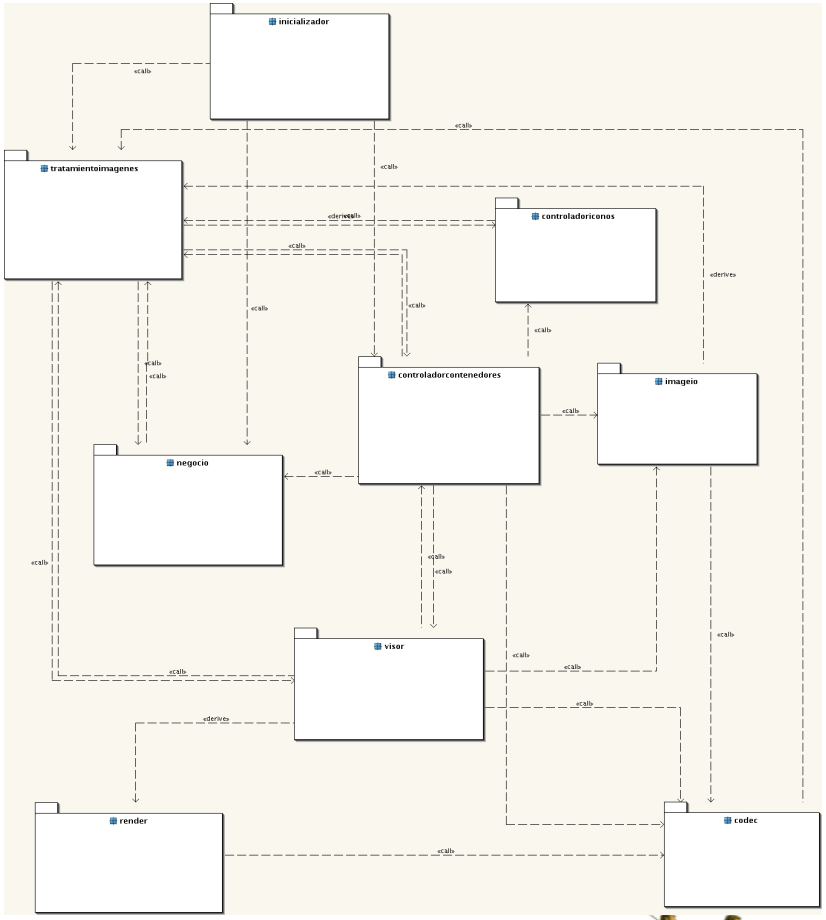


# 3. Case study: I. Architecture Recovery

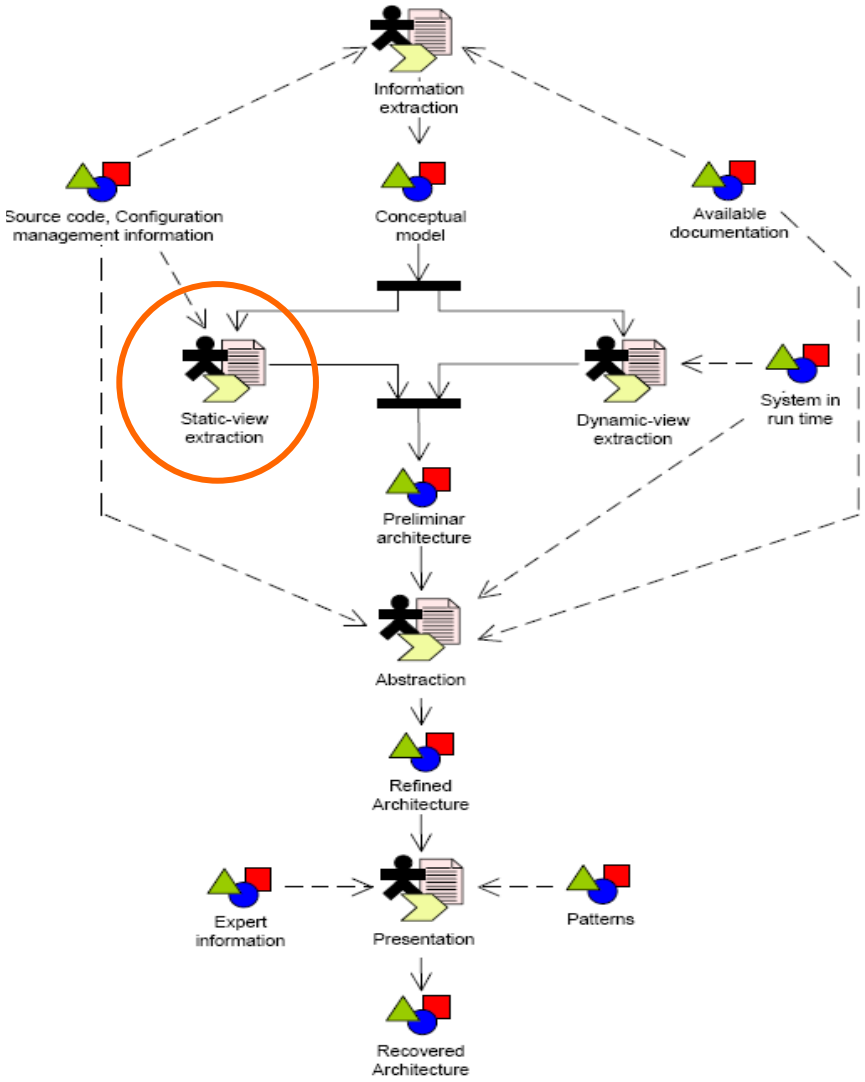



## 2. Static-view extraction. Tools:

- **Jude Community Edition**

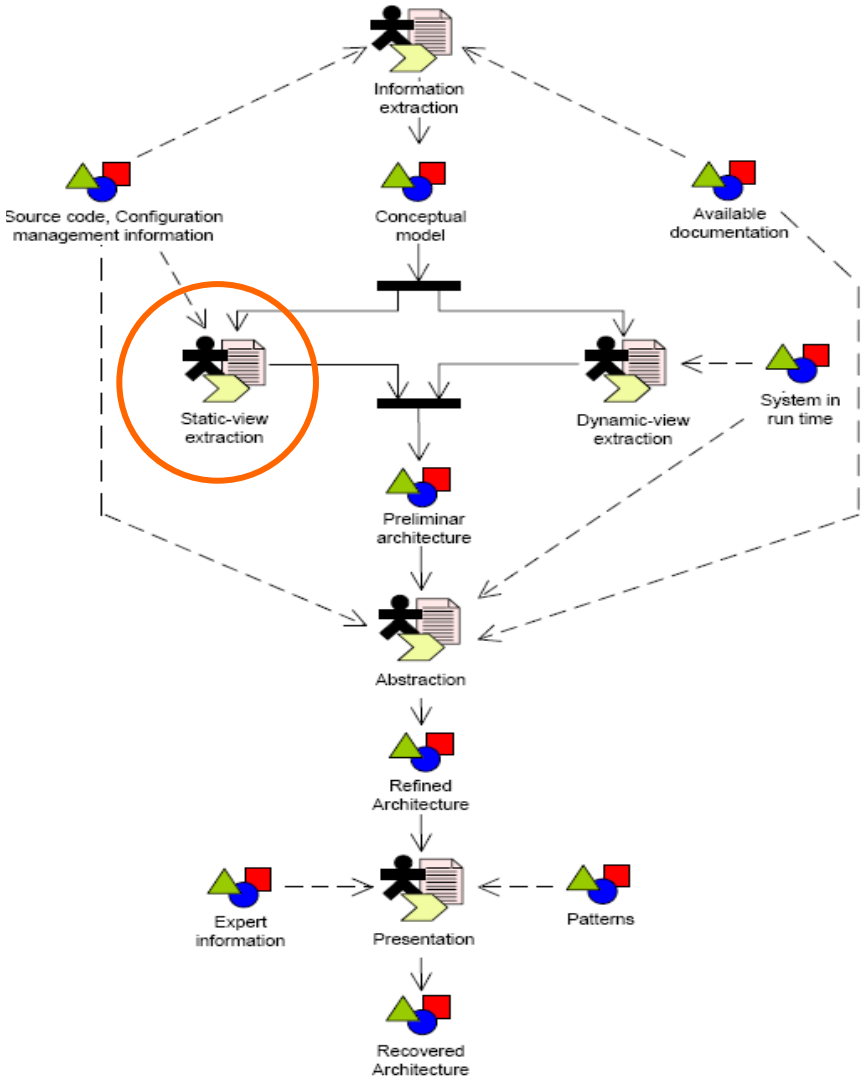



# 3. Case study: I. Architecture Recovery



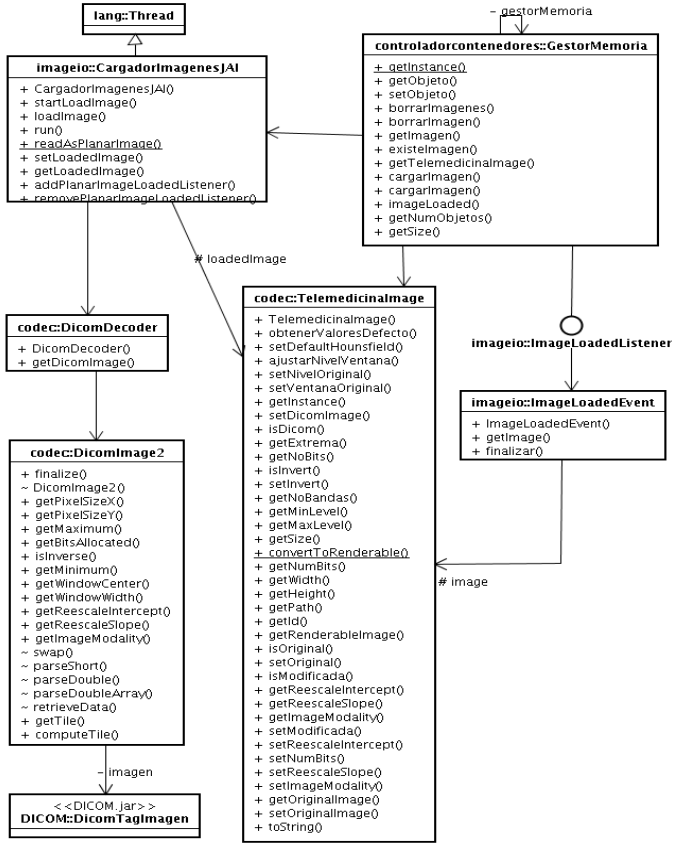

2. **Static-view extraction. Tools:**
   - **Omondo Studio Edition**
     - **Class diagrams**
     - **Association detection**

# 3. Case study: I. Architecture Recovery

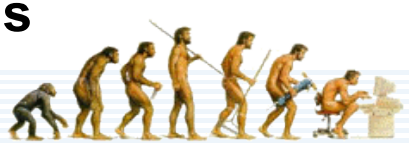



## 2. Static-view extraction. Tools:

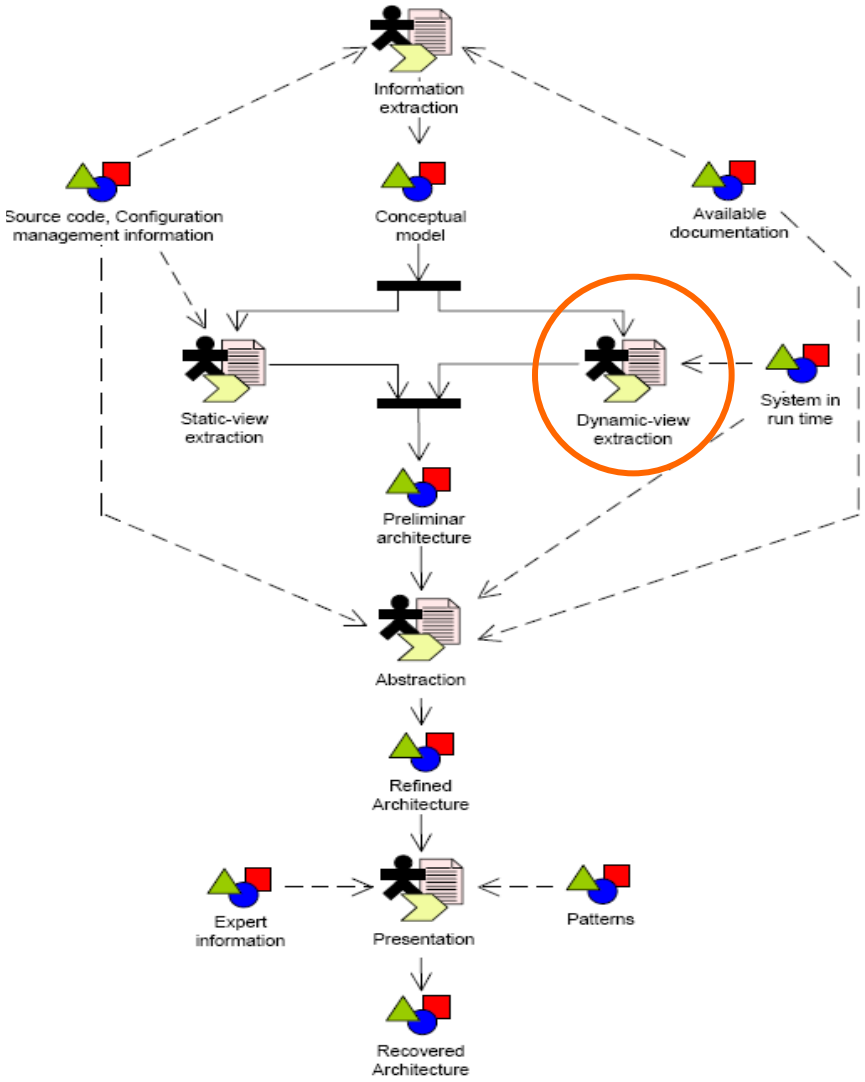
- **Omondo Studio Edition**




- **17 class diagrams**

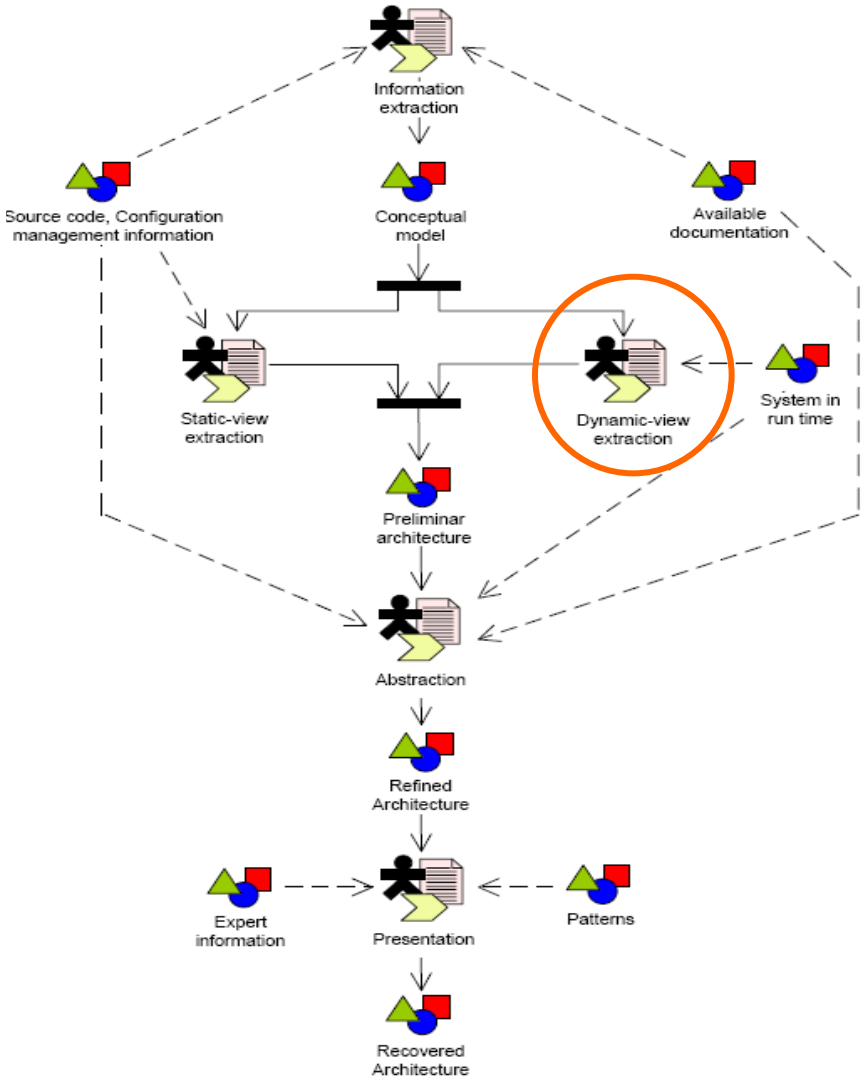


# 3. Case study: I. Architecture Recovery



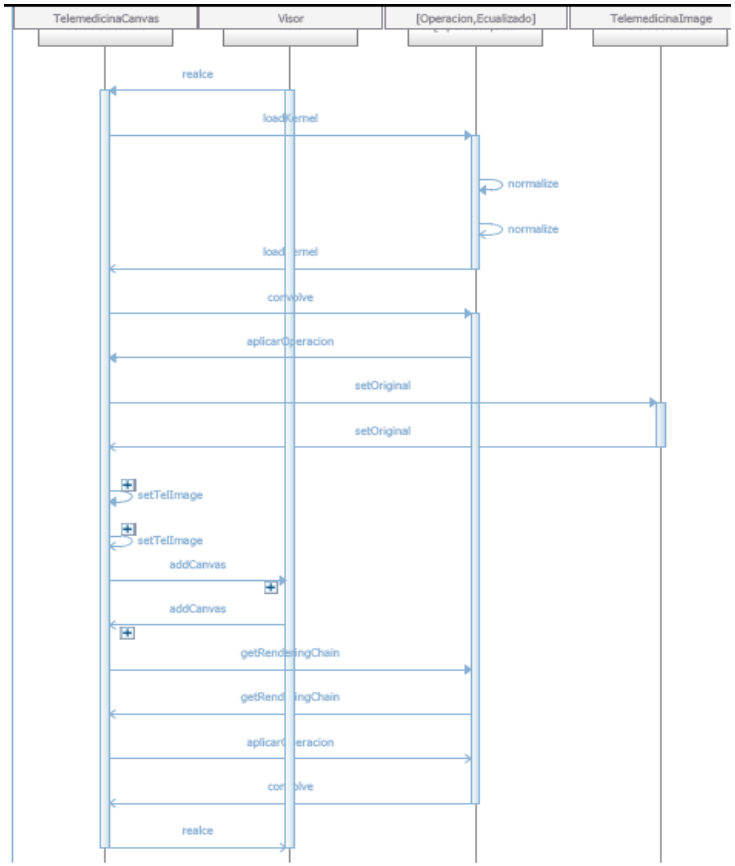

3. **Dynamic-view extraction. Tool:**
   - **Eclipse TPTP**
     - **Agents for testing and monitoring applications**
     - **Sequence diagrams**

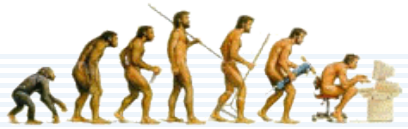
# 3. Case study: I. Architecture Recovery



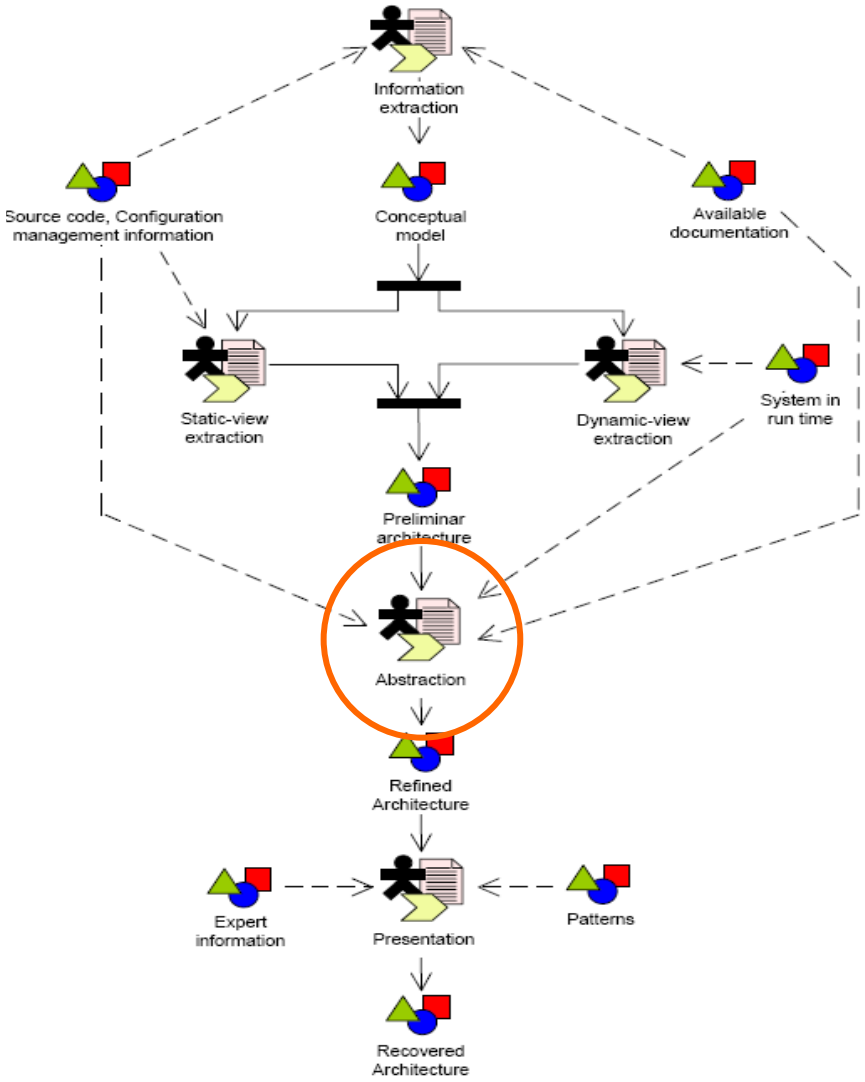

## 3. Dynamic-view extraction. Tool:

- **Eclipse TPTP**



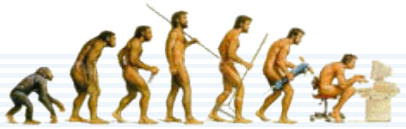



# 3. Case study: I. Architecture Recovery

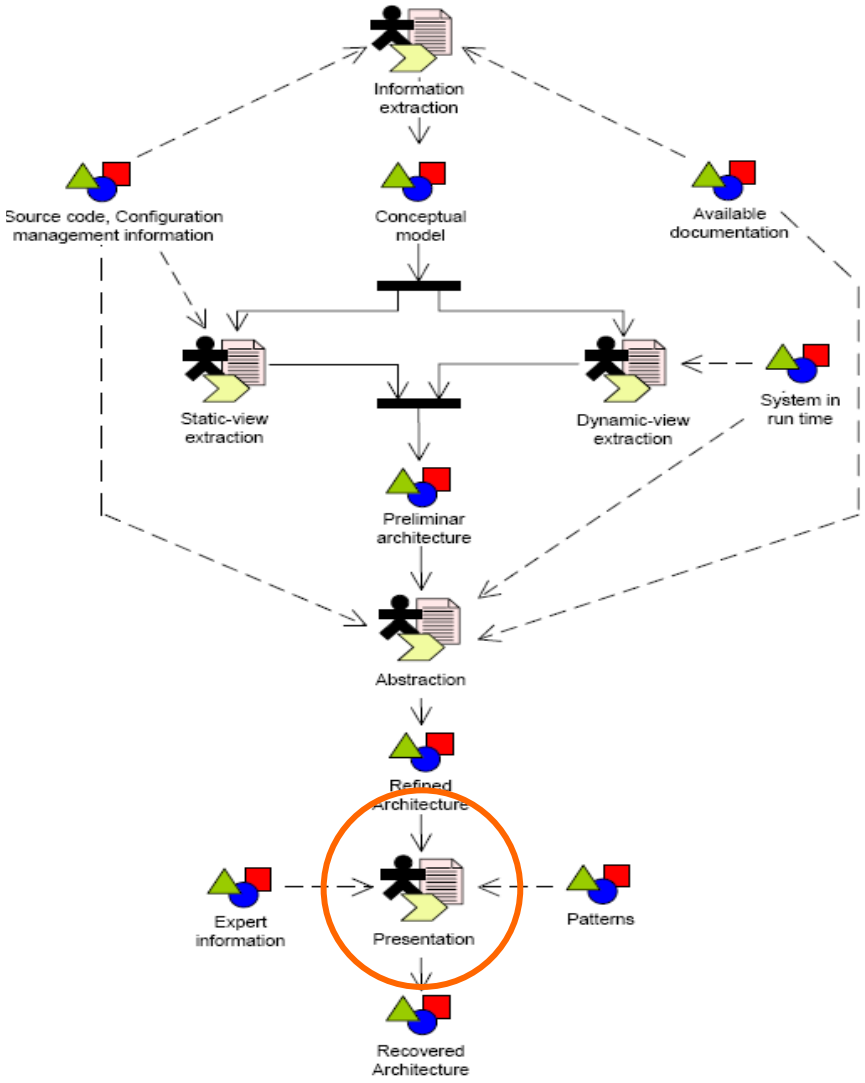



## 4. Abstraction

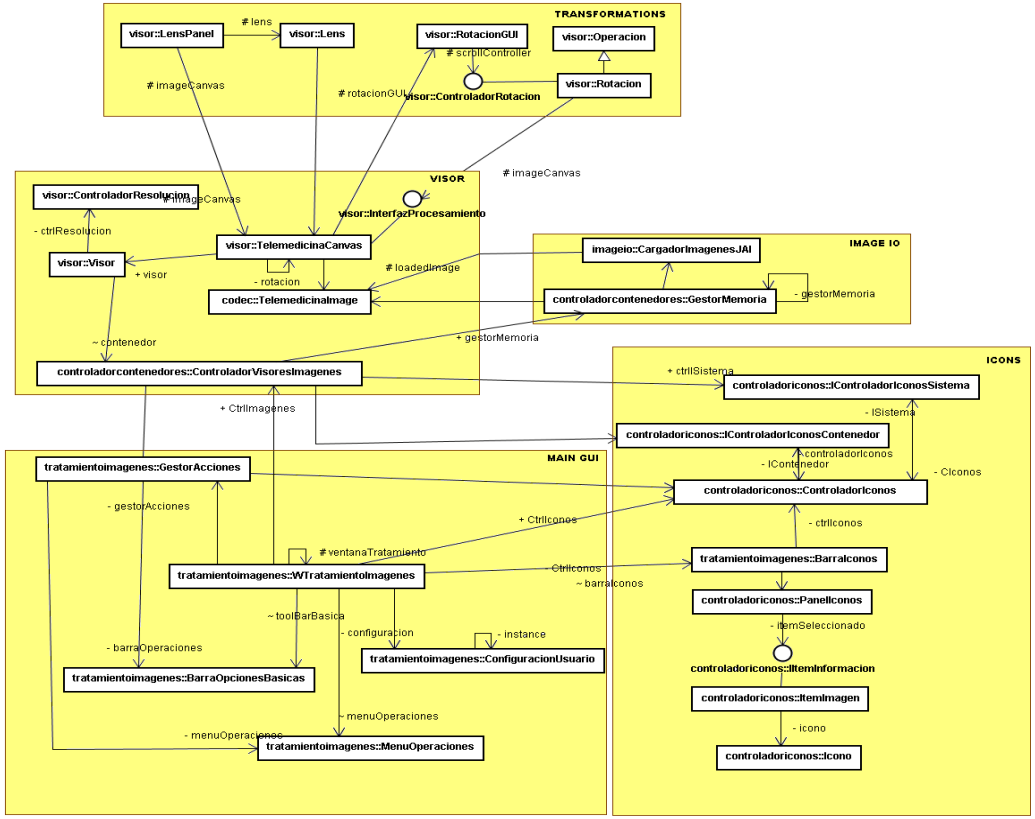
- Filter non-relevant elements
- Filter unused elements
- Detect fundamental classes
- Define higher level modules



# 3. Case study: I. Architecture Recovery




## 5. Presentation (final architecture)

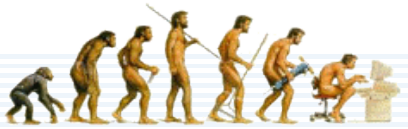



- 21 classes (90% abstraction)
- Defining higher level modules

# 3. Case study: II. Evolution Definition

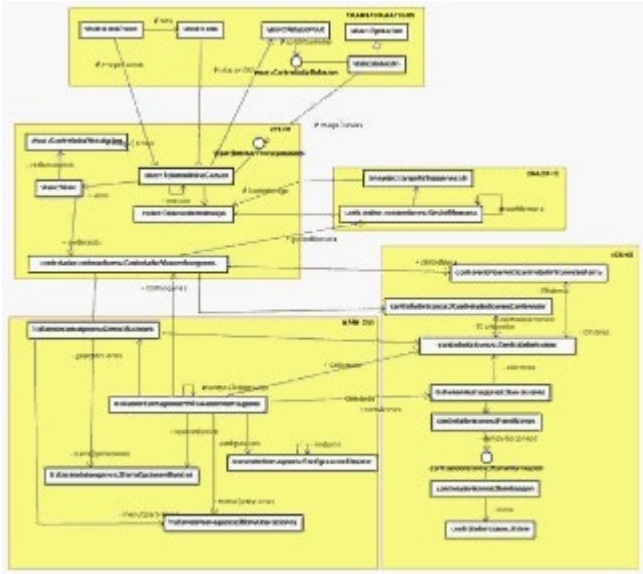
a. Architecture selection
   - Framework: OSGi
     - Service Oriented Java Framework
     - OSGi R4 implemented by Eclipse Equinox
   - Eclipse PDE (Plugin Development Environment) as workbench

b. Definition of the steps:
   - Based on architecture diagrams recovered
   - Dividing packages in bundles
   - Linking components by Whiteboard



# 3. Case study: II. Evolution Definition

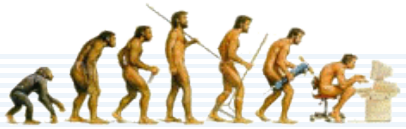
c. Planning of the steps



- Best practices: resolving dependencies with `import-package`

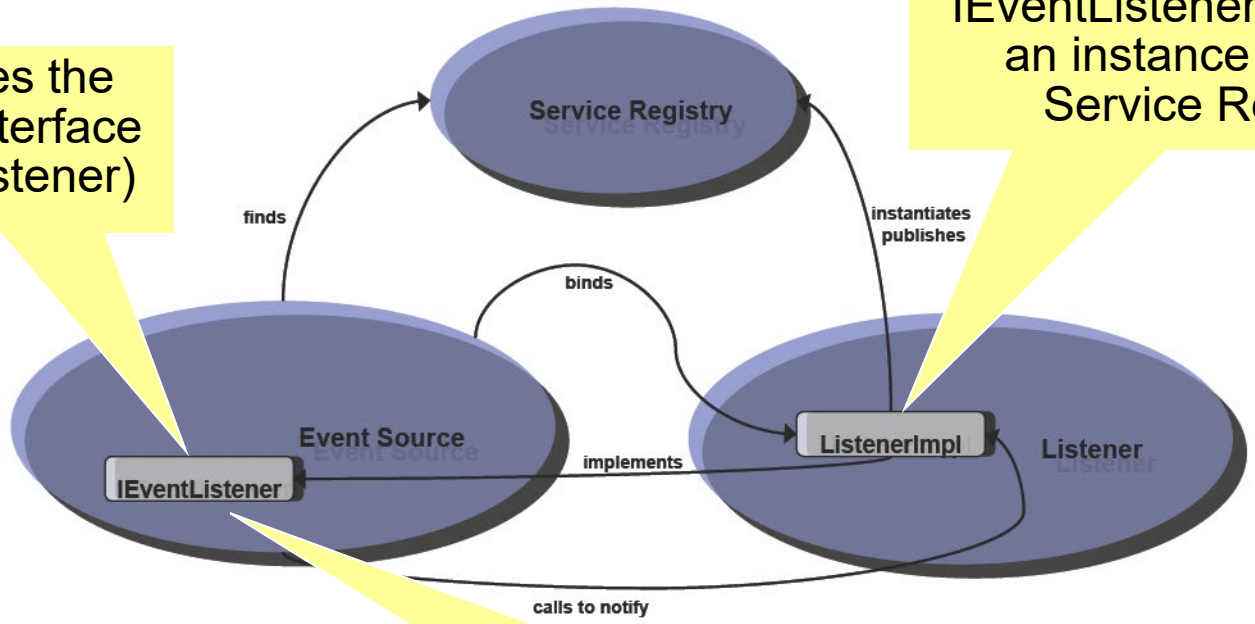
d. Feasibility check of the steps

- Unit testig in OSGi bundles with JUnit



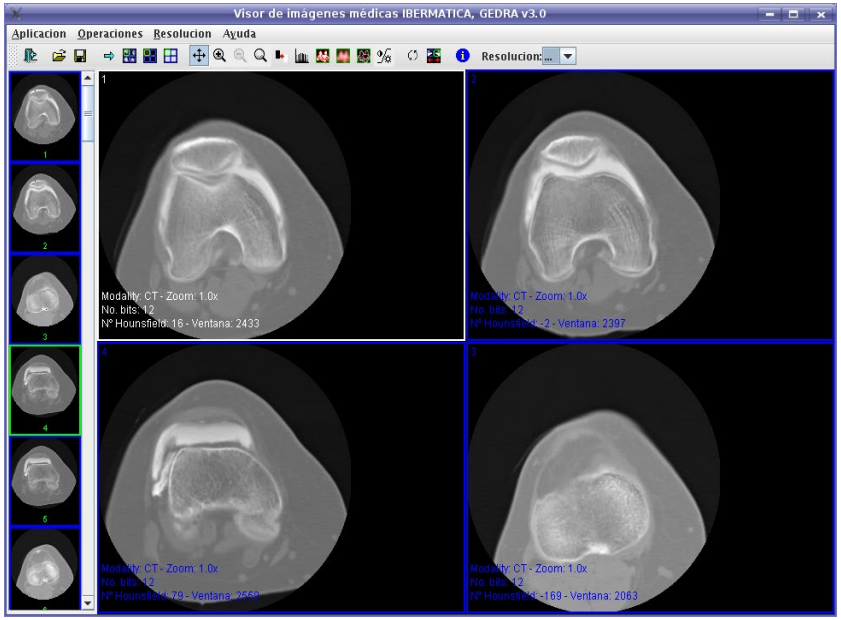
# 3. Case study: III. Evolution Execution

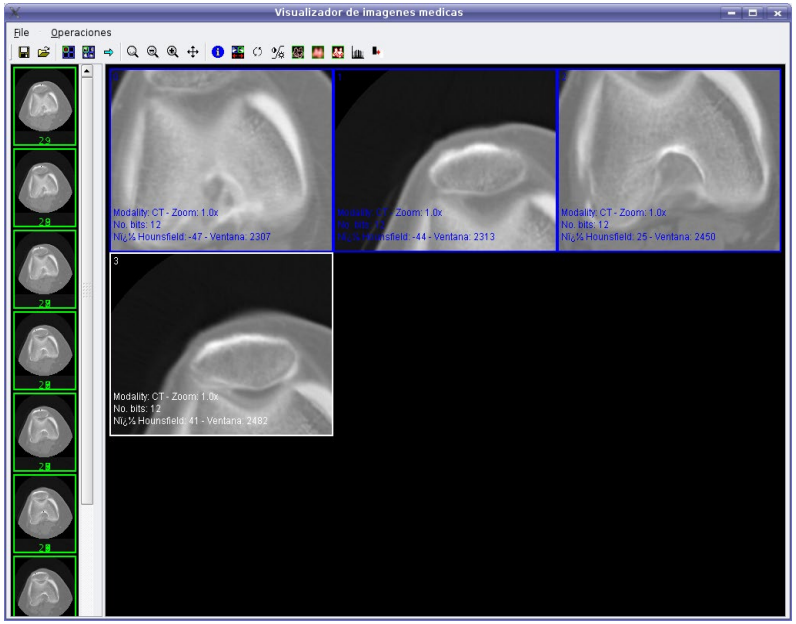
- Binding Services with *whiteboard* pattern



# 3. Case study: III. Evolution Execution

- Result: decoupling view from logic (Swing & RCP)

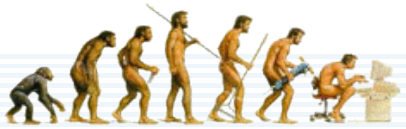



- (implemented two GUIs with the same underlying logic)

# 4. Conclusions

- Architecture recovery as 1st stage in evolution to SOA
- How to recover the architecture of a system:
  - QAR workflow
  - Java Tools:
    - Static-View: Jude & Omondo
    - Dynamic-View: Eclipse TPTP
- How to evolve a Java application to SOA-OSGi:
  - Platform: Eclipse Equinox and PDE
  - Bundles dependencies: `import-package`
  - Decoupling services: Whiteboard pattern



SOCNE 2008

# Thank you for your attention

*Boni García*
*bgarcia@dit.upm.es*

*26th March 2008, Okinawa*