An Eclipse Based Environment to Define and Execute Processes with Application to the Reverse Engineering [*]

Andrea De Lucia[1], Fausto Fasano[1], Michele Risi[1], and Giuseppe Scanniello[2]

Dipartimento di Matematica e Informatica, University of Salerno[1],

Dipartimento di Matematica e Informatica, University of Basilicata[2]

Outline

- Motivation
  - Software maintenance
  - Reverse engineering tools
  - Workflow management technologies
- The framework
  - Reusing software components
- Case study
- Conclusion and future work

Motivation

- Successful software system is forced to change over time

- Changes are needed:

  - to meet new user requirements,

  - to adapt software to interact with external entities, e.g., people, organizations, and artificial systems

  - to correct faults

  - to improve performances and quality

  - to migrate towards new technologies

The first Lehman's law [1980]:
"A program that is used in a real world environment necessarily must change or become progressively less useful in that environment".

Motivation

- Software Maintenance
  - The maintenance is the largest and the most expensive activity
    - It starts when the software product is delivered and ends when the product is no longer available for use.
  - The cost is due to technical and managerial problems
    - It increases in case methodology to anticipate the changes have been not used and the documentation is lacking
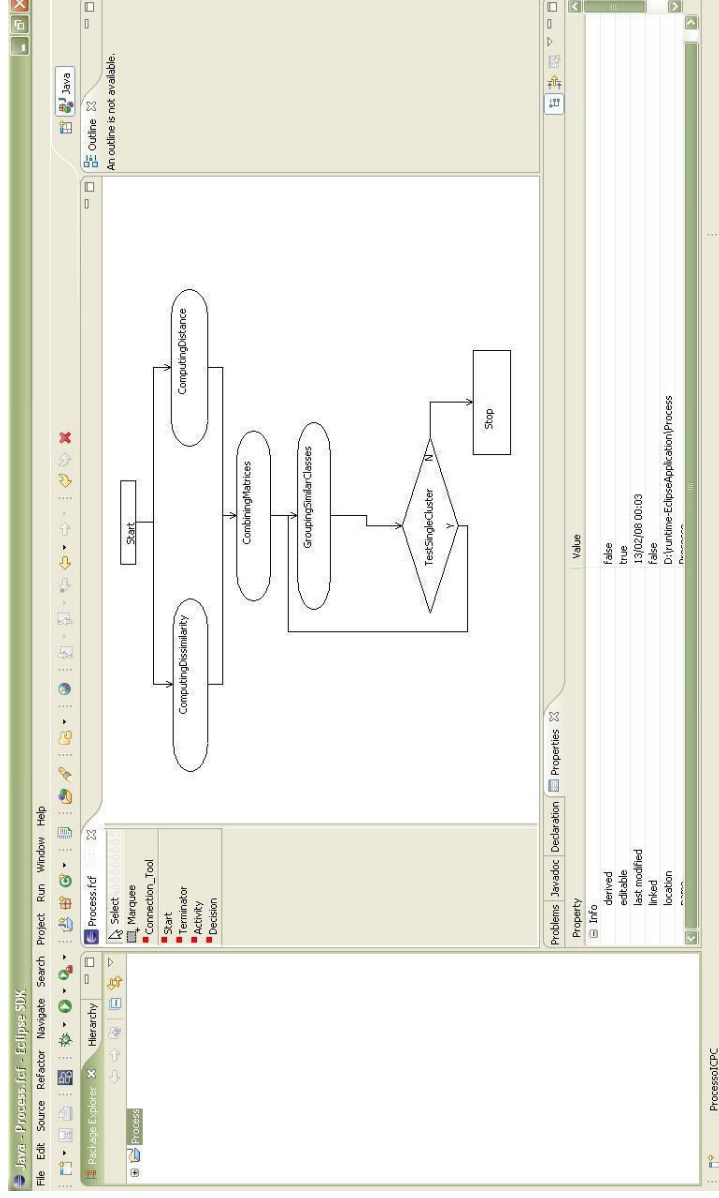
Reverse engineering

- To reduce the maintenance effort reverse engineering tools can be used
  - They support the comprehension of existing software systems, abstracting higher level models from the code (the major source of information)
- Different freeware and commercial tools are available
- All-in-one reverse engineering and comprehension tools often lack
  - A high effort is required to design and develop a specific tool
- To meet this concern tools have been proposed in the past
  - They are generally based on some domain specific or scripting language, thus requiring specific competences
  - Many of them enable the reuse of existing reverse engineering components so the const consists of integrating these components into a reverse engineering process

The visual framework

- Workflow management technologies could represent a viable solution to
  - define and execute reverse engineering and comprehension processes
  - integrate different tools and components
- We propose a visual framework
  - Eclipse plug-in
  - UML activity diagrams
  - Predefined or newly developed components can be reused
    - Implemented with traditional programming languages and software environments for data analysis (i.e., MATLAB or R)
  - The process can be partially executed
    - In case the early activities of the process have been previously executed the software engineer can decide to skip them and run however the process.
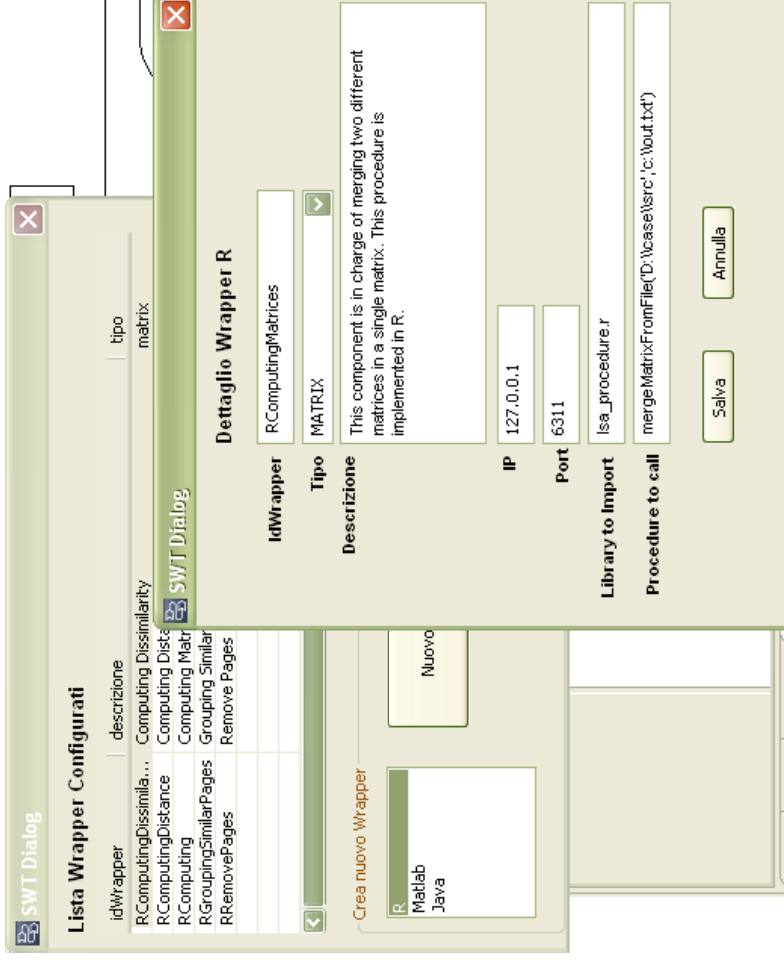
Defining a process

- Activities are arranged
- Components are associated to each activity
- Great emphasis has been provided on the reuse of software components implemented using
  - R
  - MATLAB
- Rserve is used to integrate R components
- JMatLink engine is used to integrate MATLAB components
- Visual sentences are encoded in an XML file

Associating existing components

- An easy to use mechanism is provided

- Components implemented in R, MATLAB, and Java ready to be used are available
  - Wrappers implemented once for all

- Figure shows how an existing R component is associated to an activity

```
<matrix id="RComputingMatrices" class-name="cddTools.MatrixGeneratorR" class-path="">
    <parametri-generali>
        <p-g name="IP" value="127.0.0.1" />
        <p-g name="PORT" value="6311" />
        <p-g name="PROC" value="mergeMatrixFromFile('D:\\case\\src','c:\\out.txt')" />
        <p-g name="LIB" value="source('lsa_procedure.r')" />
    </parametri-generali>
    <parametri-posizionali></parametri-posizionali>
</matrix>
```
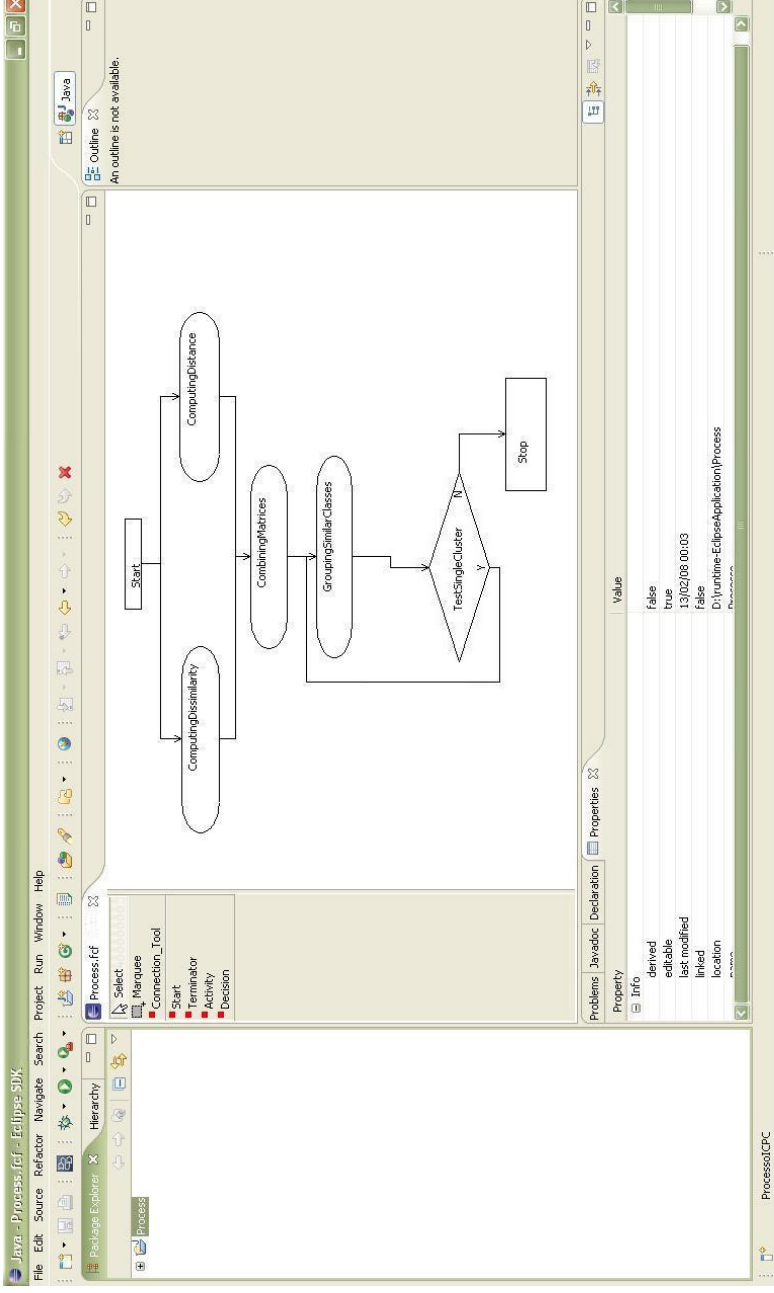
Case study

- The framework has been assessed on different processes
  - Many previously defined before implementing the framework
- In this paper:
  - A process for the software remodularization of existing objects oriented software systems
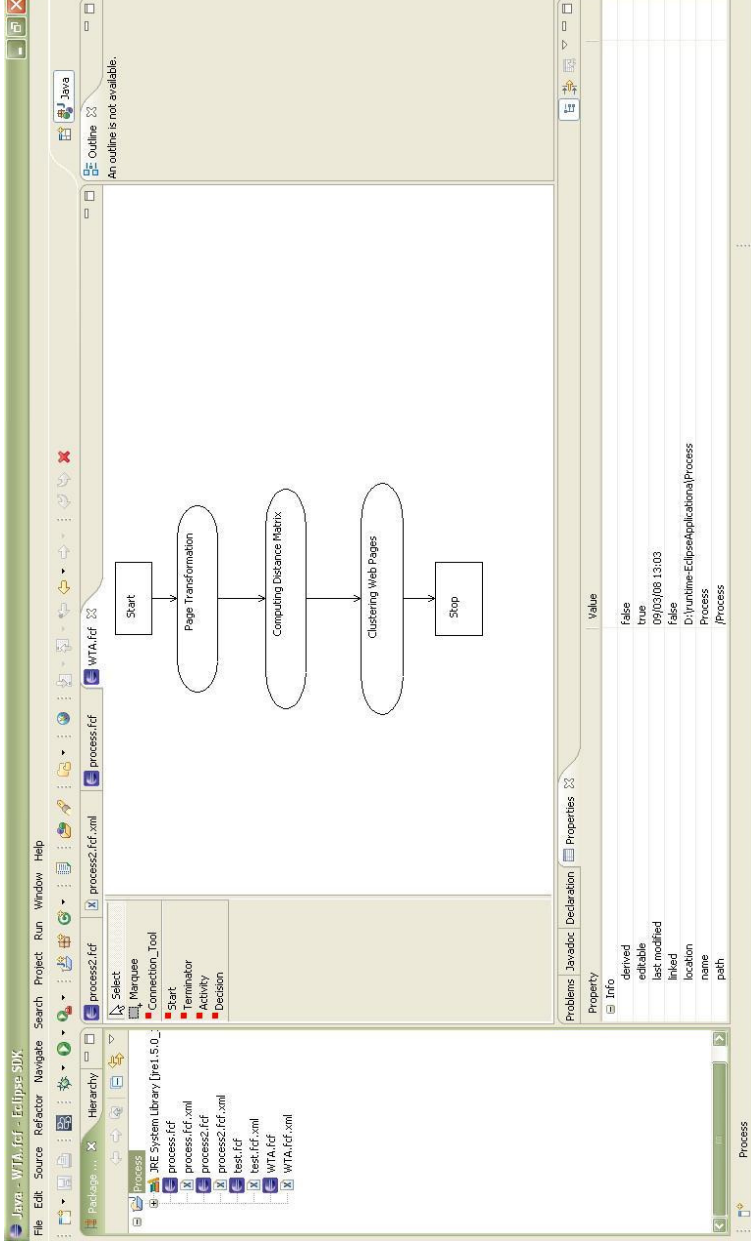 - A process to identify similar web pages

Software remodularization

- ComputingDissimilarity
  - Input: source code
  - Output: dissimilarity matrix
- ComputingDistance
  - Input: source code
  - Output: distance matrix
- CombiningMatrices
  - Input: distance and dissimilarity matrices
  - Output: combined matrix
- GroupingSimilarClasses
  - Input: combined matrix and number of clusters to identify (the k-means clustering algorithm has been used)
  - Output: groups of classes

- jEdit 4.3
  - a text editor for programmers
- JHotDraw 5.1
  - a two-dimensional graphics framework

Grouping similar pages

- Page Transformation
  - Input: static and dynamic pages of a web application
  - Output: strings encoding page structures

- Computing Distance Matrix
  - Input: strings
  - Output: distance matrix (computed using the Levenshtein algorithm)

- Clustering Web Pages
  - Input: distance matrix
  - Output: groups of similar pages (the Winner-Takes-All clustering algorithm has been used)

- The process was previously defined*
- Possible differences have been investigates

* De Lucia A., Scanniello G., Tortora G.: Identifying Similar Pages in Web Applications using a Competitive Clustering Algorithm. In. Journal on Software Maintenance and Evolution, vol. 19, no. 5, John Wiley & Sons, (2007) 281-296.

Conclusion

- A framework to define and execute processes to reverse engineering and comprehend existing software systems has been proposed

  – It has been developed as Eclipse plug-in

  – It integrates a visual environment (implemented using Graphical Editing Framework) where processes are specified in terms of UML activity diagrams

  – The definition of the process is completed associating newly developed or predefined software components

- The framework has been assessed on some case studies.

  – On processes to comprehend existing web applications and to remodularize legacy information systems

Future directions

- Future work will be devoted to further experiment the environment on different processes
- We plan to empirically validate the usefulness of the plug-in
  - Controlled experiments with students, academic researchers, and professional programmers
  - Actual industrial case studies with professional programmers
- Usability studies will be also performed

Thanks!!!

Contact information:
Giuseppe Scanniello
Dipartimento di Matematica e Informatica
Università della Basilicata
Viale Dell'Ateneo n°10,
Macchia Romana
85100, Potenza, ITALY
Email: giuseppe.scanniello@unibas.it
gscanniello@unisa.it