An approach for the evolutionary discovery of software architectures

Aurora Ramírez, José Raúl Romero, Sebastián Ventura

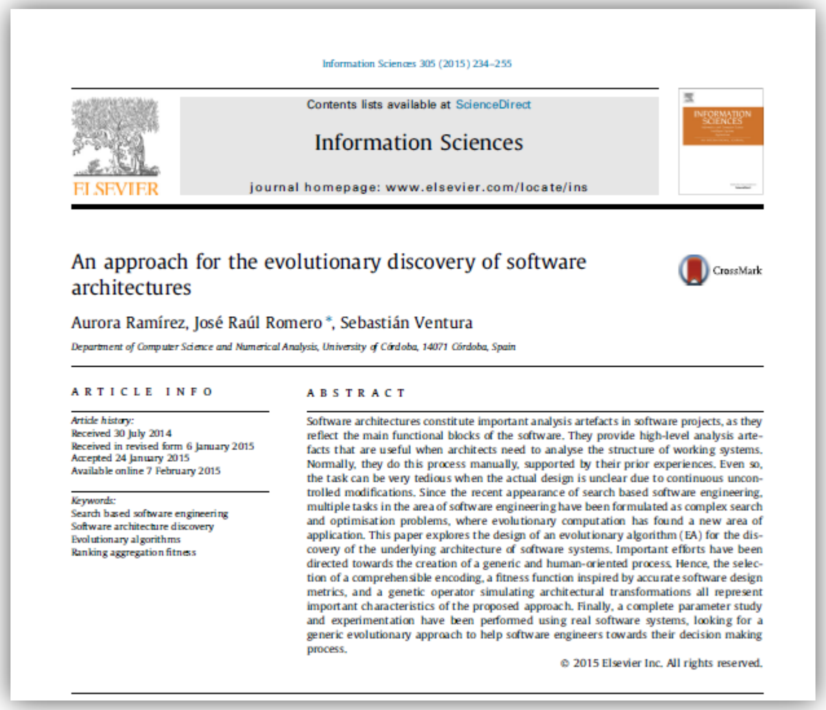
Dept. Computer Science and Numerical Analysis. University of Córdoba.

XXI Jornadas en Ingeniería del Software y Bases de Datos (JISBD)
Salamanca (Spain). 13-16 September 2016

Content

1. Introduction
2. The search problem
3. Evolutionary approach
4. Experimental study
5. Conclusions

A. Ramírez, J.R. Romero, S. Ventura. "An approach for the evolutionary discovery of software architectures". *Information Sciences*, vol. 305, pp. 324-255. 2015.


Information Sciences 305 (2015) 234–255

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

An approach for the evolutionary discovery of software architectures

Aurora Ramírez, José Raúl Romero *, Sebastián Ventura

Department of Computer Science and Numerical Analysis, University of Córdoba, 14071 Córdoba, Spain

ARTICLE INFO

Article history:
Received 30 July 2014
Received in revised form 6 January 2015
Accepted 24 January 2015
Available online 7 February 2015

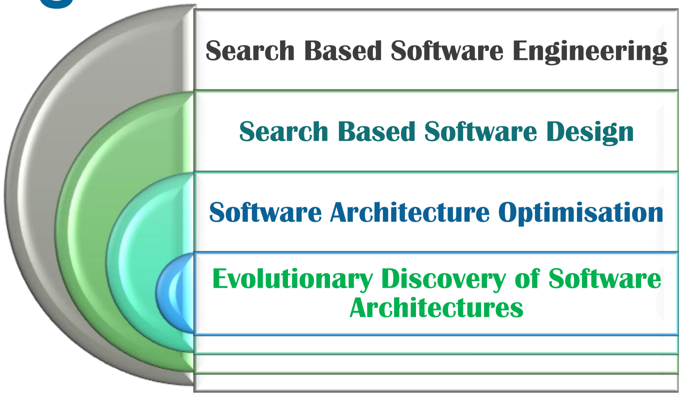
Keywords:
Search based software engineering
Software architecture discovery
Evolutionary algorithms
Ranking aggregation fitness

ABSTRACT

Software architectures constitute important analysis artefacts in software projects, as they reflect the main functional blocks of the software. They provide high-level analysis artefacts that are useful when architects need to analyse the structure of working systems. Normally, they do this process manually, supported by their prior experiences. Even so, the task can be very tedious when the actual design is unclear due to continuous uncontrolled modifications. Since the recent appearance of search based software engineering, multiple tasks in the area of software engineering have been formulated as complex search and optimisation problems, where evolutionary computation has found a new area of application. This paper explores the design of an evolutionary algorithm (EA) for the discovery of the underlying architecture of software systems. Important efforts have been directed towards the creation of a generic and human-oriented process. Hence, the selection of a comprehensible encoding, a fitness function inspired by accurate software design metrics, and a genetic operator simulating architectural transformations all represent important characteristics of the proposed approach. Finally, a complete parameter study and experimentation have been performed using real software systems, looking for a generic evolutionary approach to help software engineers towards their decision making process.

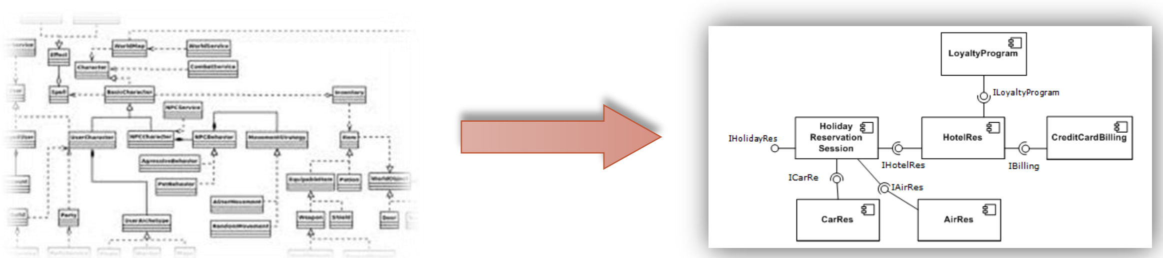
Introduction

- Software architects face complex design decisions
  - Software structure, platforms, styles...
  - Functional and non-functional requirements
  - Few information at this stage of the development
- **Search Based Software Engineering**
  - Support in decision making
  - Exploration of design alternatives



The search problem

- We want to automatically identify the underlying architecture from an analysis model (represented as a class diagram)



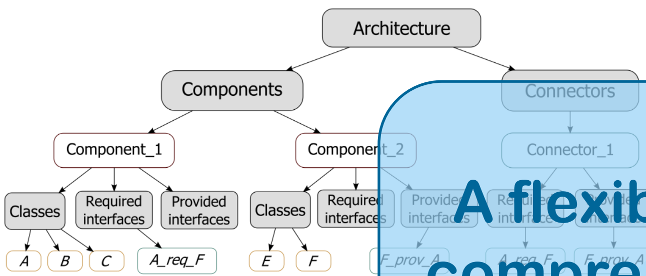
- It can be a too demanding, complex and time-consuming task
- **Evolutionary algorithms** may serve to (semi-)automate the process of finding optimal software architectures
- A extremely high combinatorial problem

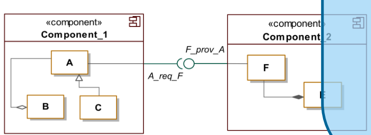
The search problem
Research questions

RQ1: Can single-objective evolutionary algorithms help the software engineer to identify an initial candidate architecture of a system at a high level of abstraction?

RQ2: How does the configuration of the algorithm influence both the evolutionary performance and the quality of the returned solution?

Evolutionary approach
Key elements





A flexible and comprehensive tree encoding

$$fitness(ind) = \begin{cases} r(ICD_{ind}) + r(ERP_{ind}) + r(GCR_{ind}) & \text{if ind is valid} \\ \#individuals \cdot \#metrics + 1 & \text{if ind is invalid} \end{cases}$$

$$ICD_i = \frac{\#classes_{total} - \#classes_i}{\#classes_{total}} \cdot \frac{CI_i^{in}}{CI_i^{in} + CI_i^{out}}$$

$$ICD = \frac{1}{n} \cdot \sum_{i=1}^{n} ICD_i$$

$$ERP = \sum_{i=1}^{n} \sum_{j=1}^{n} \left[w_{as} \cdot n_{as_{ij}} + w_{ag} \cdot n_{ag_{ij}} + w_{co} \cdot n_{co_{ij}} + w_{ge} \cdot n_{ge_{ij}} \right]$$

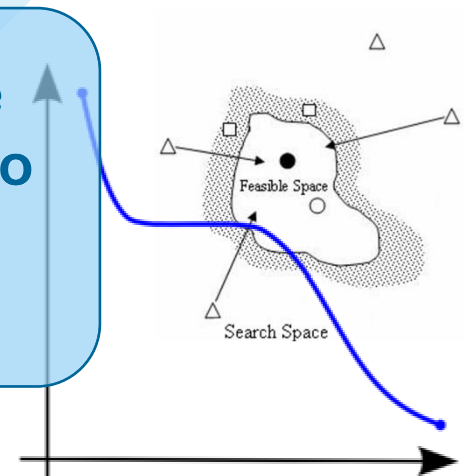
$$GCR = \frac{\#cgroups}{\#components}$$

A fitness function based on three design metrics

RQ1

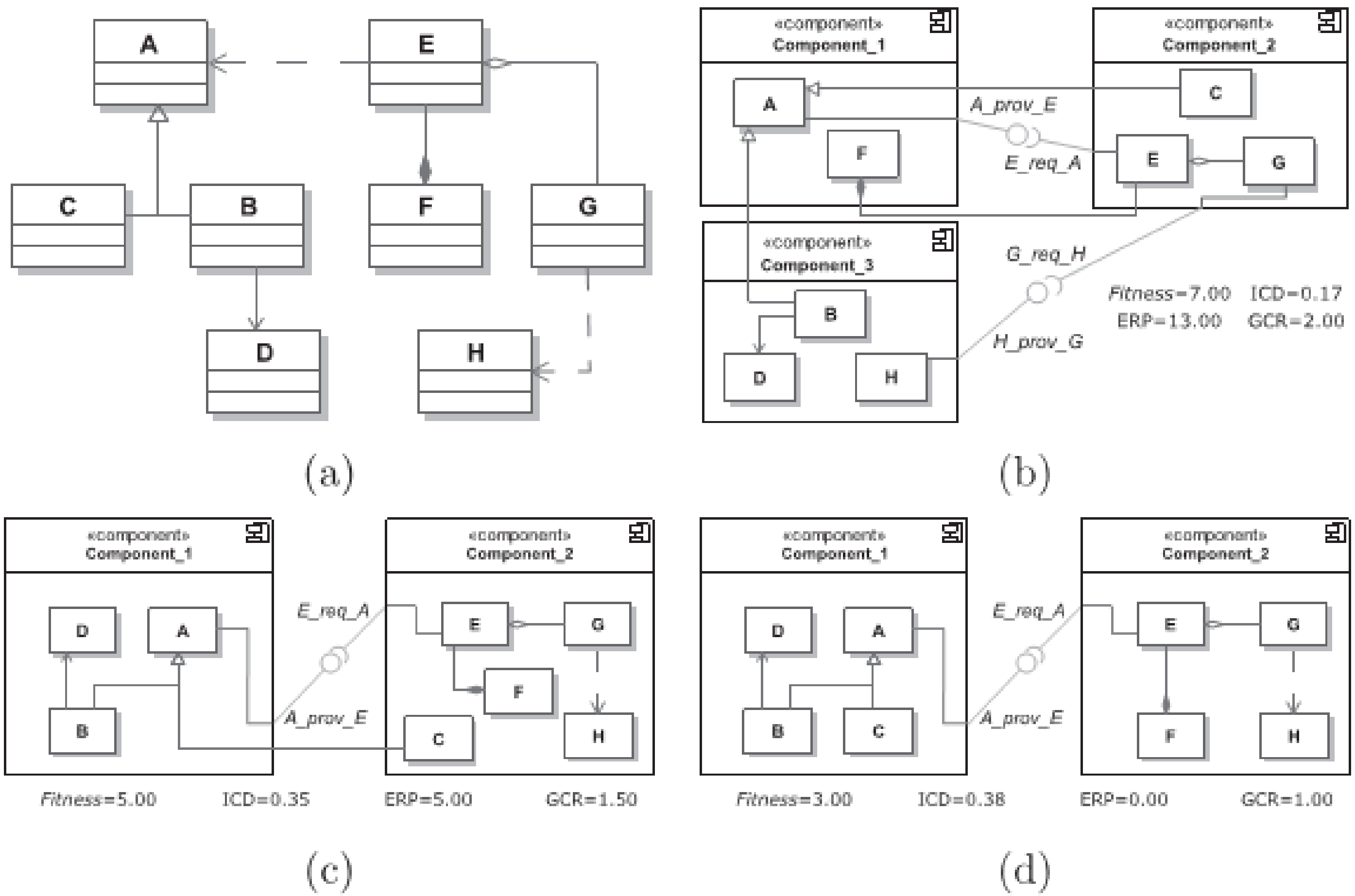
A mutation operator able to perform five architectural transformations

An adaptive mechanism to deal with constraints



Evolutionary approach
Illustrative example

RQ1



Experimental study

RQ2

Parameter study

Selection	deterministic / **tournament** / roulette
Replacement	best / competition / elitism / **elitism (10%)** / binary tournament
Mutation	[0.1,0.6] -- P_{add} = **0.2** | P_{remove} = P_{merge} = **0.1** | P_{split} = P_{move} = **0.3**
Population Size	50, 100, **150**, 200
Stopping criterion	convergence every 1200 evaluations: **20000-24000**

Experimental results

- Optimal or near optimal values for **GCR**
- ICD and ERP require to **strike a balance**
- Without assuming any structure, it can **identify related functional blocks**
- Importance of the **number and types of relationships** among classes

Conclusions

- Evolutionary Computation as **an exploratory mechanism** to decision support
  - Identify blocks of related functionality
  - Without assuming any structure
- The search approach **is close to the architect**
  - Flexible and comprehensible representation
  - Architectural transformations with heuristic information
  - Fitness function based on design metrics

An approach for the evolutionary discovery of software architectures

Thanks!

UNIVERSIDAD DE CÓRDOBA

Aurora Ramírez

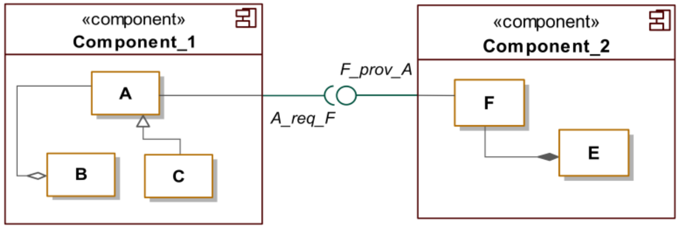
Email. **aramirez@uco.es**
Web. **http://www.uco.es/users/aramirez/en**

XXI Jornadas en Ingeniería del Software y Bases de Datos (JISBD)
Salamanca (Spain). 13-16 September 2016

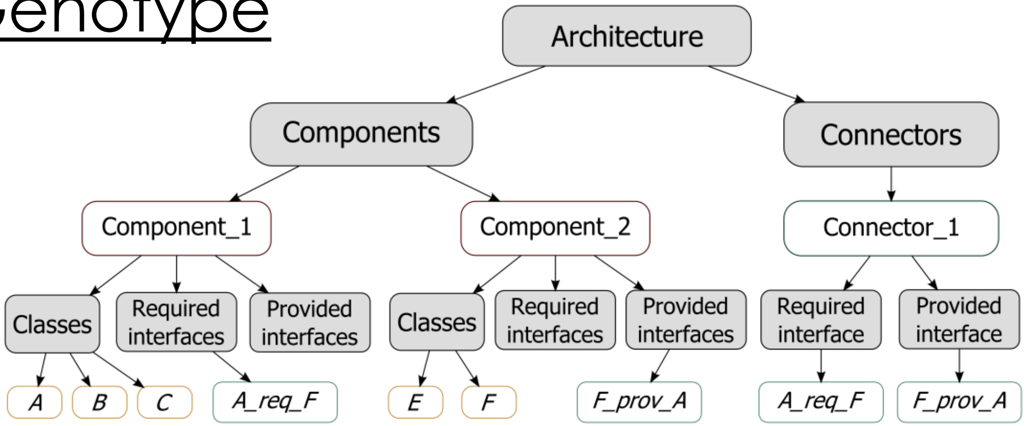
Evolutionary approach
Problem representation and constraints

Architectural solutions (individuals) are ***coded as multi-layered trees***

Phenotype



Genotype



Initialisation and constraints

1. Random distribution of classes
   ✓ No empty components and no replicated classes
2. Set interfaces and connectors
   ✗ Isolated or mutually dependant components

An approach for the evolutionary discovery of software architectures. SBSE@JISBD'16.

Evolutionary approach
Fitness function

- The fitness function is based on **three design metrics:**
  - Intra-modular coupling density (ICD)
  - External relations penalty (ERP)
  - Groups/components ratio (GCR)

$$ICD_i = \frac{\#classes_{total} - \#classes_i}{\#classes_{total}} \cdot \frac{CI_i^{in}}{CI_i^{in} + CI_i^{out}}$$

$$ICD = \frac{1}{n} \cdot \sum_{i=1}^{n} ICD_i$$

$$ERP = \sum_{i=1}^{n} \sum_{j=1}^{n} \left[w_{as} \cdot n_{as_{ij}} + w_{ag} \cdot n_{ag_{ij}} + w_{co} \cdot n_{co_{ij}} + w_{ge} \cdot n_{ge_{ij}} \right]$$

$$GCR = \frac{\#cgroups}{\#components}$$

- To compute the fitness of each individual, the **sum of its rankings** is calculated:

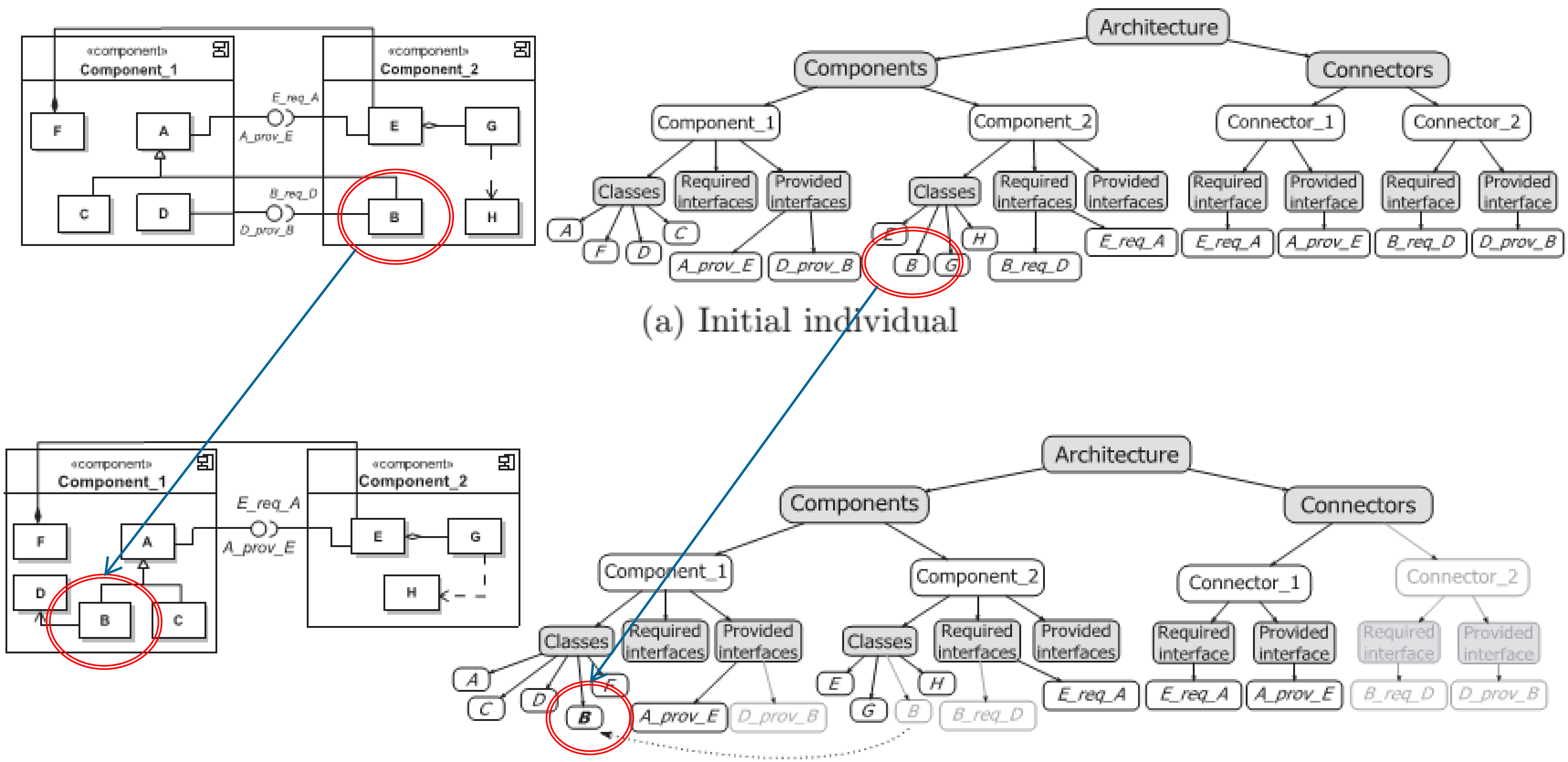
$$fitness(ind) = \begin{cases} r(ICD_{ind}) + r(ERP_{ind}) + r(GCR_{ind}) & if\ \ ind\ is\ valid \\ \#individuals \cdot \#metrics + 1 & if\ \ ind\ is\ invalid \end{cases}$$

An approach for the evolutionary discovery of software architectures. SBSE@JISBD'16.

Evolutionary approach
Genetic operator

- A mutation operator simulates the **architectural transformations** (as a software architect would do):
  - ✓ Add a component
  - ✓ Remove a component
  - ✓ Merge two components
  - ✓ Split a component
  - ✓ Move a class (random)
- **Probabilistic roulette** for each parent
- **Infeasible individuals** are controlled

Evolutionary approach
Genetic operator



(a) Initial individual

(c) Move a class mutation procedure

An approach for the evolutionary discovery of software architectures. SBSE@JISBD'16.