Generating architecture models using genetic algorithms

Kai Koskimies
Number of students: ~12000
Teaching & Research staff: ~2000
Annual Budget: ~65ME + ~65ME

Faculties:
- Automation, Mechanical and Materials Engineering
- Built Environment
- Business and Technology Management
- **Computing and Electrical Engineering**
- Science and Environmental Engineering
Faculty of Computing and Electrical Engineering

Departments

Electrical Engineering
Electronics and Communications Engineering
Signal Processing
Pervasive Computing
Department of Pervasive Computing

Personnel:
- 10 professors
- Total of about 140 professors/teachers/researchers/research assistants

Output/year (estimated)
- ~15000 credit points
- ~60 MSc’s
- 5-6 PhD’s
Laboratories

**Software Engineering** (prof. Kai Koskimies, prof. Kari Systä)
- Process, project management, product/configuration management, testing, specification and design methods, OO Methods, software architectures...

- Design and implementation of embedded systems, operating systems...

**Distributed Software** (prof. Tommi Mikkonen, department head)
- Implementation of distributed systems, Internet application technology, mobile applications...

**Usability** (prof. Kaisa Väänänen-Vainio-Mattila, prof. Timo Saari)
- Usability, user experience, UI design, ...

**Information Security** (prof. Jarmo Harju)
- Secure programming, secure networking, ...
Can systems design systems?

Darwin: project funded by Academy of Finland 2009-2011

Research problem: To what extent can systems design systems?

Approach: Application of genetic algorithms to synthesize a software architecture proposal from given requirements

Applications: automating software design, MDA transformations, software development work allocation planning, self-adaptive systems
Software Architecture

Here software architecture means:

UML class diagram with
• classes (& interfaces)
• operations
• attributes
• stereotypes
• use relationships
• inheritance relationships
• implementation relationships
Genetic algorithms in architecture synthesis

Refined use cases
- Functional requirements

Basic initial functional decomposition
- Null architecture (UML model)

Genetic Algorithm
- Fitness
  - Mod
  - Eff
  - Com
- Chromosome
  - Crossover
  - Mutations
- Solution base

Proposed software architecture (UML model)

Design patterns:
- Server, Façade, Mediator, Strategy, Adapter, Template, Dispatcher, Interface

Modifiability
Efficiency
Complexity

Quality requirements
Mutations: Example (Adapter)
Genetic software architecture synthesis

Team 1:
- size 20

Team 2:

Efficiency: 25
Modifiability: 50
Simplicity: 10

Applying patterns (e.g. Messaging, Adapter, Proxy, Facade, etc., also allocating component to a team)
Simulated evolution of software architectures

Initial population: random pattern applications
Mutation & crossover
Selection: next generation
Result: Best of the last generation
For a tool demo, see video at www.cs.tut.fi/~kk/DarwinDemo.wmv
Genetic model synthesis in MDA
Using architecture synthesis in self-adaptive systems

Motivation

• The usage environment may change during the lifetime of a software product
• Adaptive maintenance should be done by the system itself as much as possible
• In general, systems should be more architecture-aware, capable of reflecting and improving their own architecture

Genetic software architecture synthesis provides an approach for pattern-based self-adaptive systems, capable of architecture level self-improvement
Self-adapting architecture infrastructure

Based on Javeleon (Java platform supporting run-time changing of classes)
Reflections

Domain model + Patterns = Architecture (structure)

Architecture + Logic = Application (behavior)
References


References


