EMPAT

Fifth Special Track on Evolvable Modularity Patterns

PATTERNS 2021

Introduction
Prof. dr. Herwig T. Mannaert
University of Antwerp
Modularity and Engineering

• Modularity is a cornerstone of engineering:
  - Herbert Simon stated in 1962 that the \textit{architecture of the artificial is hierarchy}
  - Powerful technique in software, electronics, mechanics, construction, et cetera

• Several benefits are attributed to modularity:
  - Lower the complexity
  - Reuse components
  - Increased flexibility/evolvability

• Achieving these benefits is \textit{not straightforward}
Modularity and Engineering

• Modular coupling, dependencies and interactions should be studied and minimized:
  - *Herbert Simon* presented general analysis in 1962
  - *Dave Parnas* presented specific analysis in 1972 for software and presented his *double dictum*
  - *Carlyss Baldwin* and Kim Clarck analyzed the conundrum in 2000 and presented *design rules*

• Nevertheless, how to design modular structures in practice, is all too often unclear
  - Cross-cutting concerns may create ripple effects
  - It is a complex and multi-dimensional problem
An Inconvenient Truth

• Lack of evolvability may hamper the desired adaptability of information systems and therefore the *agility* of companies
  - Law of increasing complexity (Manny Lehman)

• Lack of evolvability hampers *adaptability* and *potential to innovate* in administrative systems
  - Companies reporting in multiple GAAP face huge ripple effects due to changes in financial rules
  - University faculties face huge transition effects when making changes to education programs
An Inconvenient Truth

• Lack of evolvability hampers scalability and sustainability of man-made artifacts
  - Modifying a house/road may create many ripple effects due to cross-cutting utility concerns
  - Requiring cleaner combustion engines leads to demolition of cars instead of engine replacements
  - Replacing a gear handle of a bike may be impossible without replacing the whole bike

• Lack of evolvability is a fundamental issue, related to scalability and sustainability of all man-made artifacts and systems
Toward Evolvable Engineering

- We scientists should strive to provide more guidance to engineers, in order to design systems with higher levels of evolvability

- This scientific guidance:
  - Is part of *design science* (~ R. Buckminster Fuller)
  - May consist of general and/or domain specific *design rules and/or theories*
  - Will probably be consolidated in generic and/or domain-specific *design patterns*

- *We scientists should strive to establish a discipline of evolvable modularity*
Imagine Evolvable Engineering

- We could change the grid architecture for electricity distribution, without impacting the internals of houses
- We could extend/rearrange buildings and roads, including utility services, without facing ripple effects in structures
- We could replace car engine(s) (parts) to comply with emission standards, without sending the entire car to the junkyard
- We could adapt financial reporting to changing regulations and reporting standards, without causing ripple effects
- We could reform and modernize education programs and tracks, without facing ripple and transition effects
- We could maintain and improve information systems, without writing them off due to large maintenance costs
- We could improve and insert description and regulation paragraphs in thousands of documents, without duplications
Imagine ...

The sciences of the evolving and growing artificial
A Special track on Evolvable Modularity Patterns (EMPAT) was first organized in April 2017 in Athens, Greece. Over the years, contributions have been presented proposing and analyzing patterns for evolvable modularity covering a wide range of application areas including housing utilities, logistics, documents, data analysis models, software code generation, and information security. In this fifth edition in April 2021 in Porto, Portugal, three papers are presented.
• Usage of Iterated Local Search to improve firewall evolvability
  *Geert Haerens*

• Pattern-Based Ontological Transformations for RDF Data using SPARQL
  *Marek Suchánek, Robert Pergl*

• Exploring the Use of Code Generation Patterns for the Creation of Evolvable Documents and Runtime Artifacts
  *Herwig Mannaert, Gilles Oorst, Koen De Cock, Peter Uhnak*