Challenges in Developing a Software Architecture Evolution Tool as a Plug-In

Jeffrey M. Barnes and David Garlan
Institute for Software Research
Carnegie Mellon University

Slides and paper available for download at:
Background

• *Architecture evolution* is central to software development

  - new technologies
  - new frameworks
  - new requirements
  - new market opportunities

  require architectural change

• At present, software architects have few tools to help plan and execute evolutions
Background

J. M. Barnes, D. Garlan, et al.

O. Le Goaer, D. Tamzalit, et al.

L. Grunske

N. Brown et al.
Our Model of Architecture Evolution

Initial architecture

Interim states

Path

Target architecture

Time
Our Previous Tooling Work

• Previously* developed a partial prototype as part of a case study at NASA JPL
• Case study aim was to model an evolution at JPL, using modeling languages and tools in use there
• Developed facilities to support modeling architecture evolution in MagicDraw
  – Representational conventions for modeling architecture evolution in UML
  – Transformation macros to effect evolution operators
  – UML models of particular intermediate states

Vision: A Plug-In for Architecture Evolution Planning
Vision: A Plug-In for Architecture Evolution Planning

Operator Palette
- Add adapter
- Configure firewall
- Remove database

Initial \(\Delta\)
Int1 \(\Delta\)
Int2 \(\Delta\)
Int3 \(\Delta\)
Int4 \(\Delta\)
Target \(\Delta\)

«transition»

Jeffrey M. Barnes and David Garlan
Development Process

• Development was carried out by a team of two master’s students, directed by the authors

• Prototype development proceeded in stages designed to produce “feature prototypes”
Feature Prototypes

1. Model manipulation
2. Identification of evolution paths
3. Interface for applying operators
4. Operator parser
5. Metadata handling
6. Positioning of presentational elements
7. Evolution path constraints

Final phase: Integration
Challenge:

Understanding the Kinds of Variation that the Framework Supports

Framework

Plug-in
Challenge:
Controlling the User Interface
Challenge:
Manipulating Presentational Elements
Conclusion

• Learned a number of lessons about factors that can simplify or complicate development of an architecture evolution plug-in
• Many of these lessons have broader application
  – Especially for other tools that must compare and analyze multiple software architectures
• Still lots of future work to do on tooling issues for architecture evolution