Detecting and Preventing the Architectural Roots of Bugs

Lu Xiao
College of Computing & Informatics Drexel University

Research Questions:

- Are buggy files architecturally connected?
- Are architecture issues the root causes of bugginess?
- Why buggy files remain buggy?

What is Missing in State-of-the-art:

- History-based defect prediction: If existing bugs are good predictor of future bugs, it means that old buggy files are never completely fixed.
- Structure-based defect prediction: not all files with structural problems are high-maintenance

Research Objective: Exploring the Architecture Roots of Error-proneness and Change-Proneness

- Viewing architecture issue as one kind of technical “debts”
  - They propagate errors among large-number of files, generating high bug rate and/or high change rate, i.e. the “interests” or “penalty” of the debt
  - Files will remain buggy if the roots remain; “interests” will accumulate as long as debts remain.

Novel Approaches:

- A novel architecture model: Design Rule Space (DRSpace)
  - Distinguish design rules and independent modules
  - Model structural and evolutionary relation simultaneously.

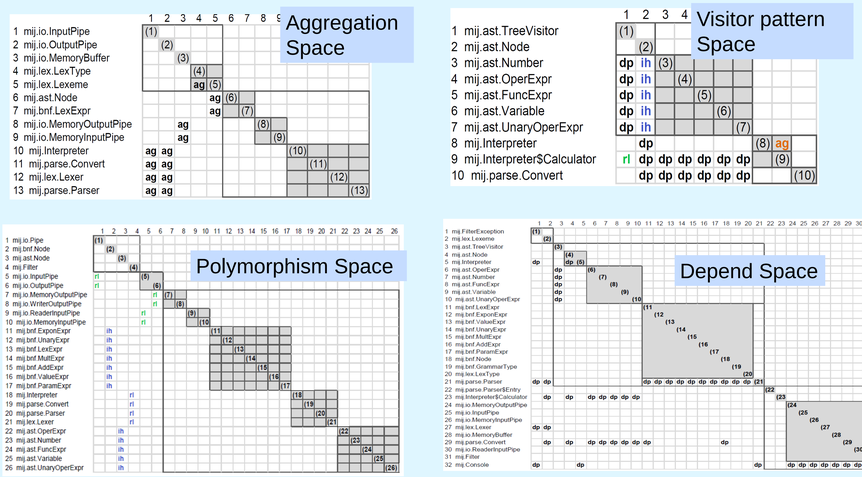

Figure 1: Four Design Rule Spaces within the Same Software System

- Modeling software architecture as multiple overlapping DRSpaces
  - Each dependency type forms a design space
  - Each design pattern forms a design space
- Architecture Root Detection
  - Calculating the interaction of design spaces and change-prone, error-prone files
- Exploring the nature of Roots
  - How do they impact error-proneness, change-proneness over time
  - How architecture issues within roots propagate errors

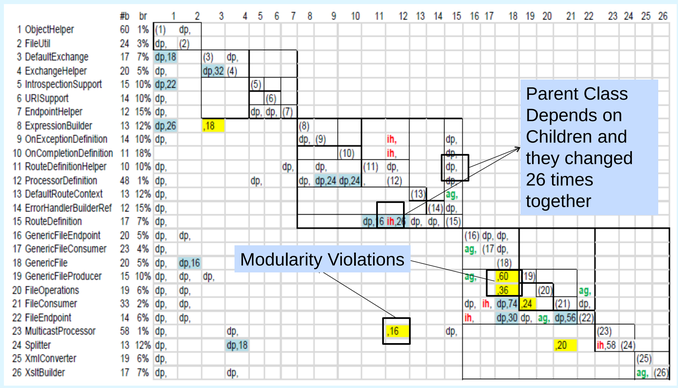

Figure 2: An Architecture root with multiple issues

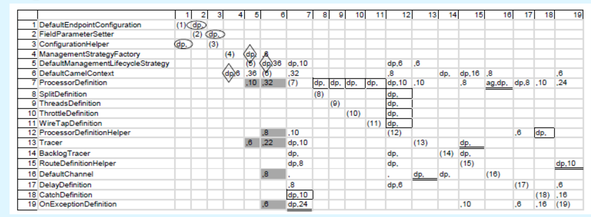

Figure 3: Cyclic Dependencies and Modularity Violations

Project	DRSpace Leading Files	Equation	R-2
Camel	CamelContext	b=0.5v+118	0.86
	Exchange	b=2.3v+82.3	0.79
PDFBox	COSName	b=5v+92.6	0.93
CXF	LogUtils	b=1.2v+170	0.88
HBase	Bytes	b=6.8v+30.6	0.83

Figure 4: How the number of buggy files grow with time

Results and Contributions:

- The impact of architecture roots are significant and persistent
  - Hundreds of buggy files can *always* be captured by just a few architecture roots, regardless of their domain, age, being open source or not.
- DRSpaces lead by error prone files also tend to be error prone.
- Error prone DRSpaces usually contain multiple architecture issues.

Implications:

- How defects are discovered, examined, and handled should be changed fundamentally:
  - Treat buggy files as architecturally connected groups.
  - Examine the architecture issues within each architecture root.
  - Reducing maintenance costs by removing architecture debts

Industrial Impact:

The supporting tool, Titan, has been used to detect architecture debts in multiple major industrial organizations.