The Looming Complexity Catastrophe in Large (and Ultra-Large) Systems

Carliss Y. Baldwin
Harvard Business School

August 15, 2005
IT system of a large retailer, 1999
Boxes=Systems; Lines=Standards
Another appalling hairball from banking—
3-year plan to do merger integration
Goals for an Ultra-Large System

- Complete interoperability of components (mix-and-match)
- Seamless upgradability
- Unlimited extensibility

No First or Second Level Hairballs!

Internet is good at the First Level but has Second Level Problems (e.g. Windows)
To avoid complexity catastrophes in Ultra-Large Systems we need:

- Tools/techniques to map actual interdependencies in large, changing codebases
- Formal measures of design option value ex ante and ex post

“If you can’t measure it, you can’t manage it.”
Complexity = Interdependency

- There are *ex ante* architectural representation languages (UML)
  - Like architectural drawings
- But very few code maps
  - Maps are *ex post*, updatable, show what’s really there
  - Like plumber’s/electrician’s diagrams
  - Example: Rusovan, Lawford, Parnas (2005) critique of Linux was based on ONE sourcefile!
- Architecture of functionally similar codebases can be very different
Browsers

Mozilla Before Redesign

Mozilla After Redesign
Word Processors

Abiword

Open Office Word
Spreadsheets

Gnumeric

Open Office Calc
These architectural differences affect design option value

- Design options have “technical potential”, denoted $\sigma$
- Technical potential, $\sigma$, varies
  - By system
  - By module
  - Over time
Evidence of Option Value

- Successive, improving versions are evidence of option values being realized over time—after the fact
- Designers “see” option values before the fact

\[ \sigma = \text{Low} \quad \text{Medium} \quad \text{Zero} \quad \text{High} \]
Measuring Option Value

◆ Measure unpredicted residuals from performance data
◆ Example: TPC-C is a benchmark for transactions processors
  – 153 submissions from 1/1/01 - 1/25/05
  – Use regression to cull out predictable performance:

\[ \text{TpmC} = a_0 + a_1(\text{time}) + a_2(\text{CPUs}) + a_3(\text{FrEnds}) + \varepsilon \]
Measuring Option Value (cont)

\[ T_{pmC} = a_0 + a_1(\text{time}) + a_2(\text{CPUs}) + a_3(\text{FrEnds}) + \varepsilon \]

\( \text{StdDev}(\varepsilon) \) is a proxy for \( \sigma \) —

\[ \sigma = \pm 124814 \]

\[ \sigma = \pm 69\% \text{ of Mean } T_{pmC} \]
To avoid complexity catastrophes in ULSs

- We need to track the evolution of architectures in real systems as they grow.
- For this we need good maps of real systems at the level of code,
- And better ways to measure option value ex ante and ex post.