Carnegie Mellon

institute for SOFTWARE RESEARCH

Talking About Concerns . . .

James D. Herbsleb
School of Computer Science
Carnegie Mellon University

Carnegie Mellon

What is Modularity?

- Thanks, Mary!
- Thanks, Dick!

institute for SOFTWARE RESEARCH

Why Modularity?

- **Software** modularity does not matter
- . . . at all
- Except . . .
  - To the extent it modularizes **work**
- **Work** modularity aids human understanding
- **Work** modularity simplifies coordinating people and teams

Parnas:
Expected Benefits of Modularity

- Reduce need for coordination
  - “separate groups would work on each module with little need for communication”
- Simplify comprehension
  - “it should be possible to study the system one module at a time”
- These effects lower the cost of change
  - “it should be possible to make drastic changes to one module without a need to change others”

Parnas, D. L. On the Criteria to be Used in Decomposing Systems into Modules. *Communications of the ACM*, 15, 12 (1972), 1053-1058, p. 1054.

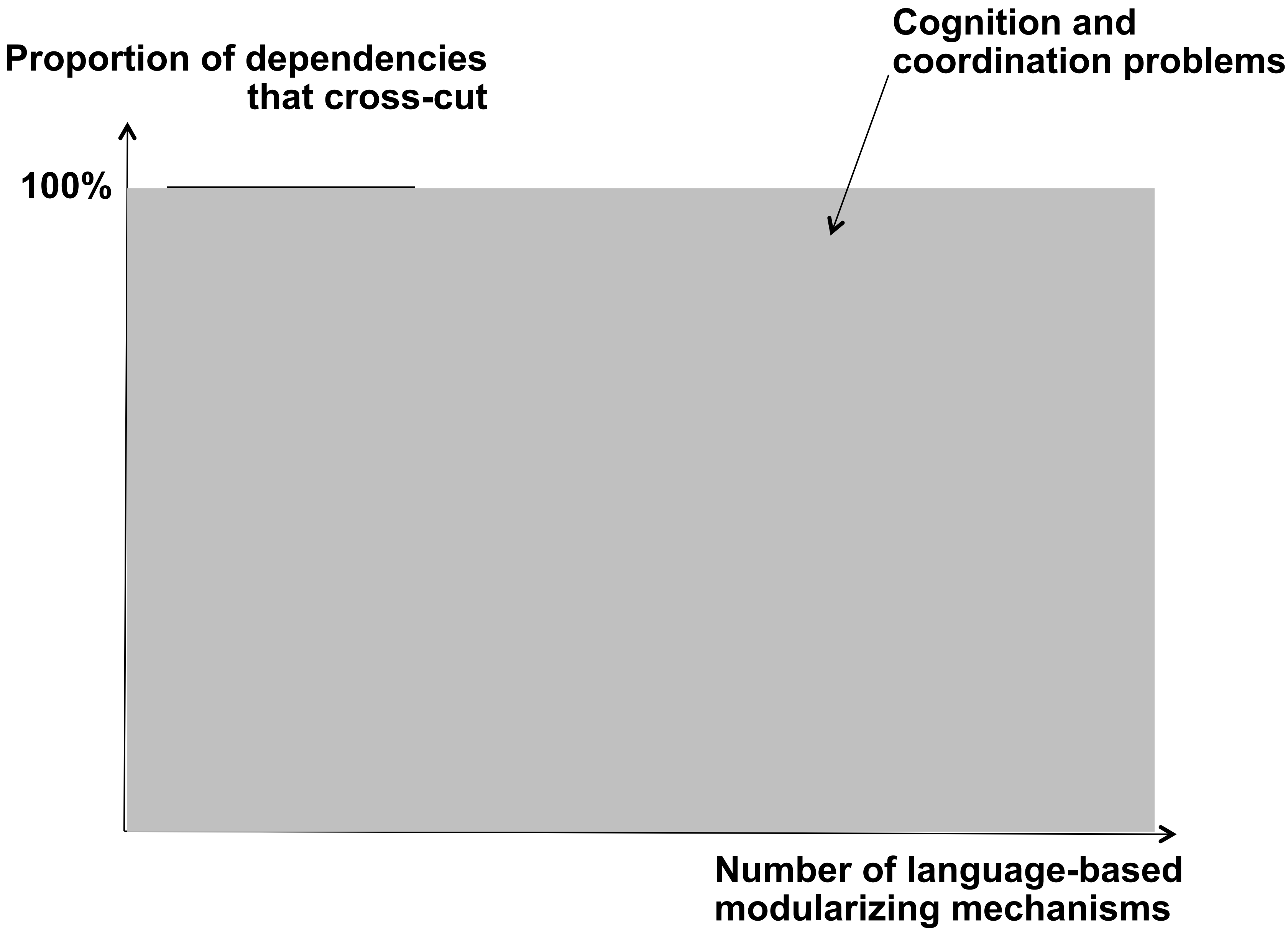
Carnegie Mellon

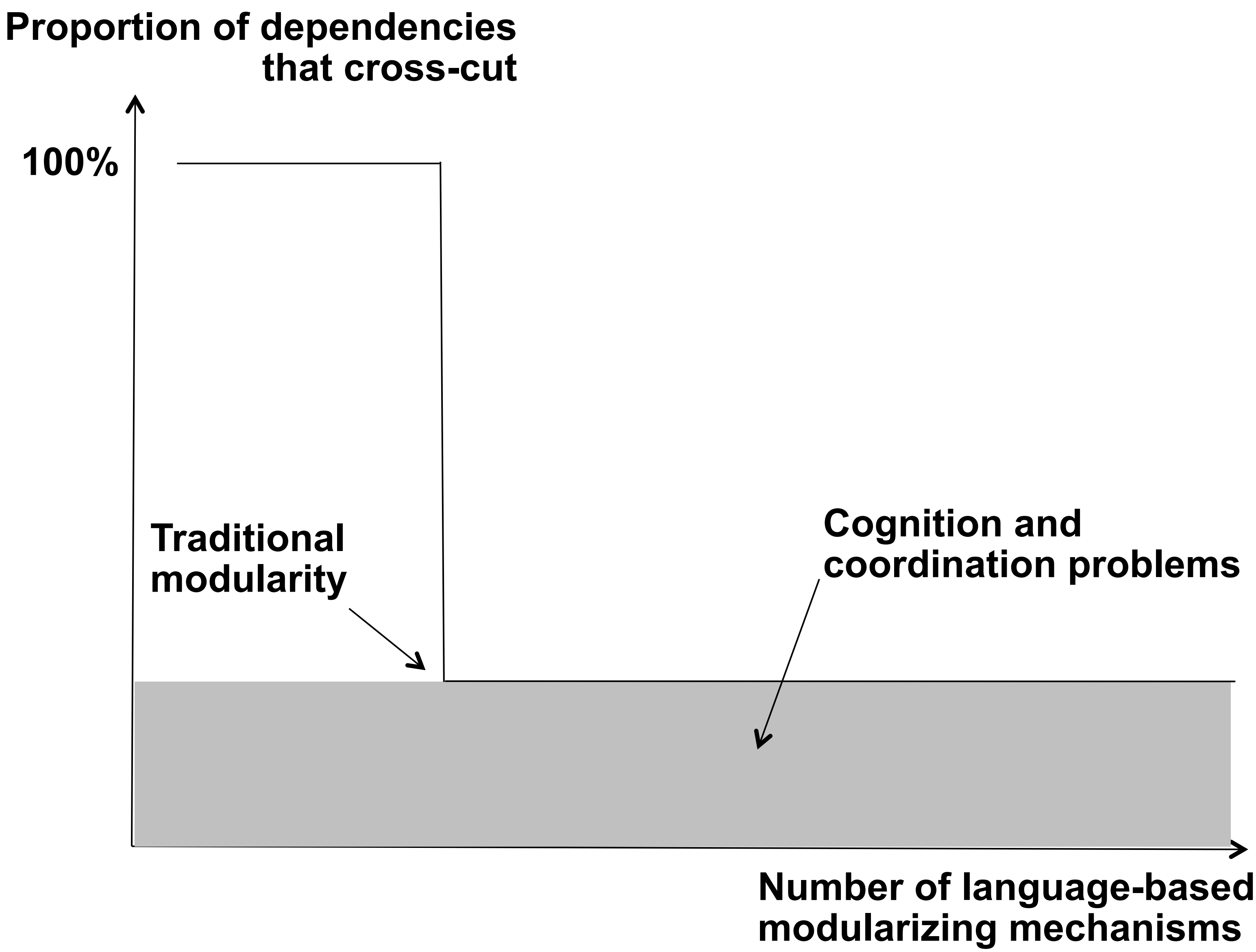
Vision . . .

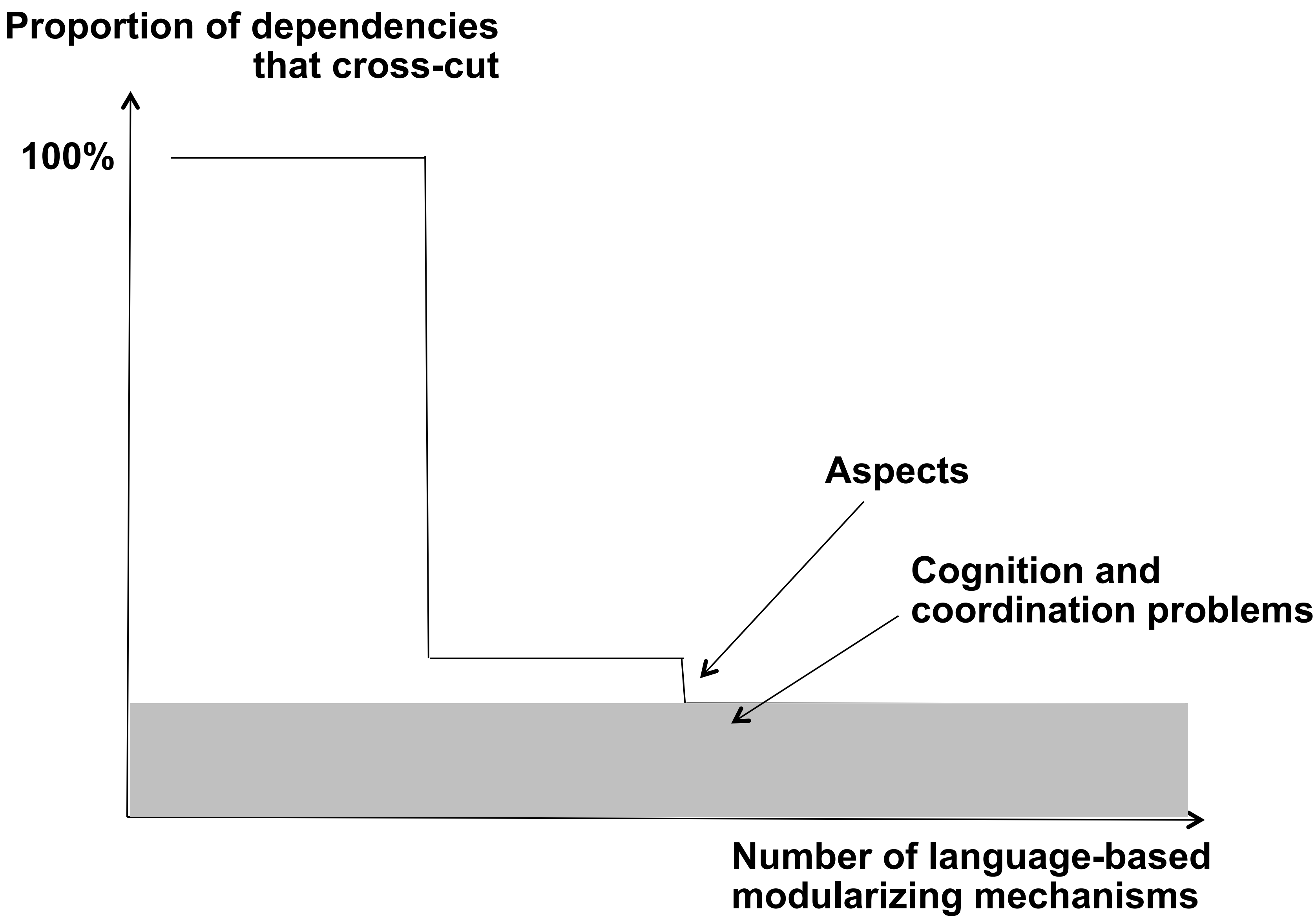
- “a vivid mental image; ‘he had a vision of his own death’” *
- “an Explanation of Life Founded upon the Writings of Giraldus and upon Certain Doctrines Attributed to Kusta Ben Luka” *
- “a thought, concept, or object formed by the imagination” **
- “direct mystical awareness of the supernatural“ **

*wordnetweb.princeton.edu/perl/webwn
**Merriam-Webster Dictionary

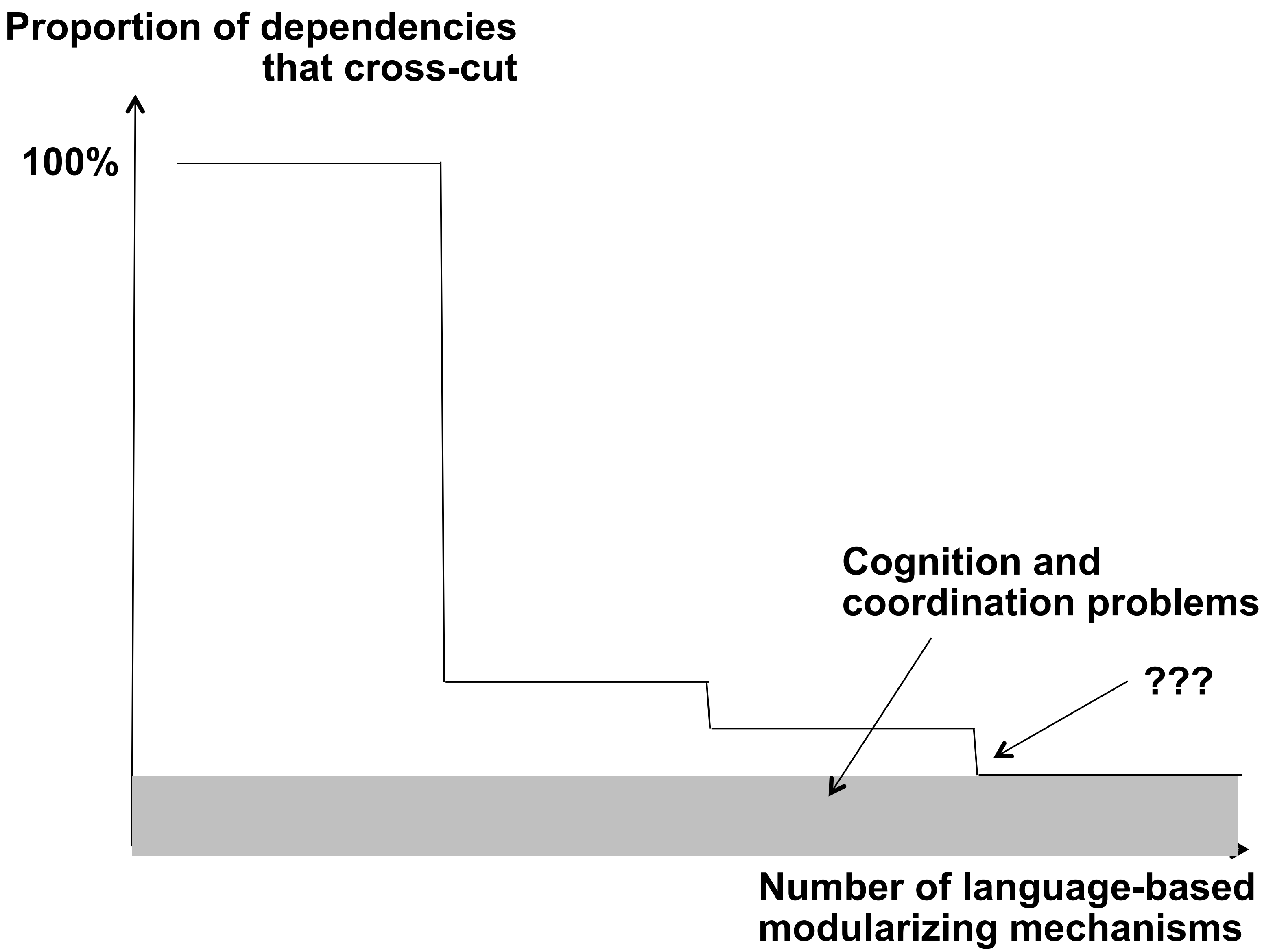
isr institute for SOFTWARE RESEARCH

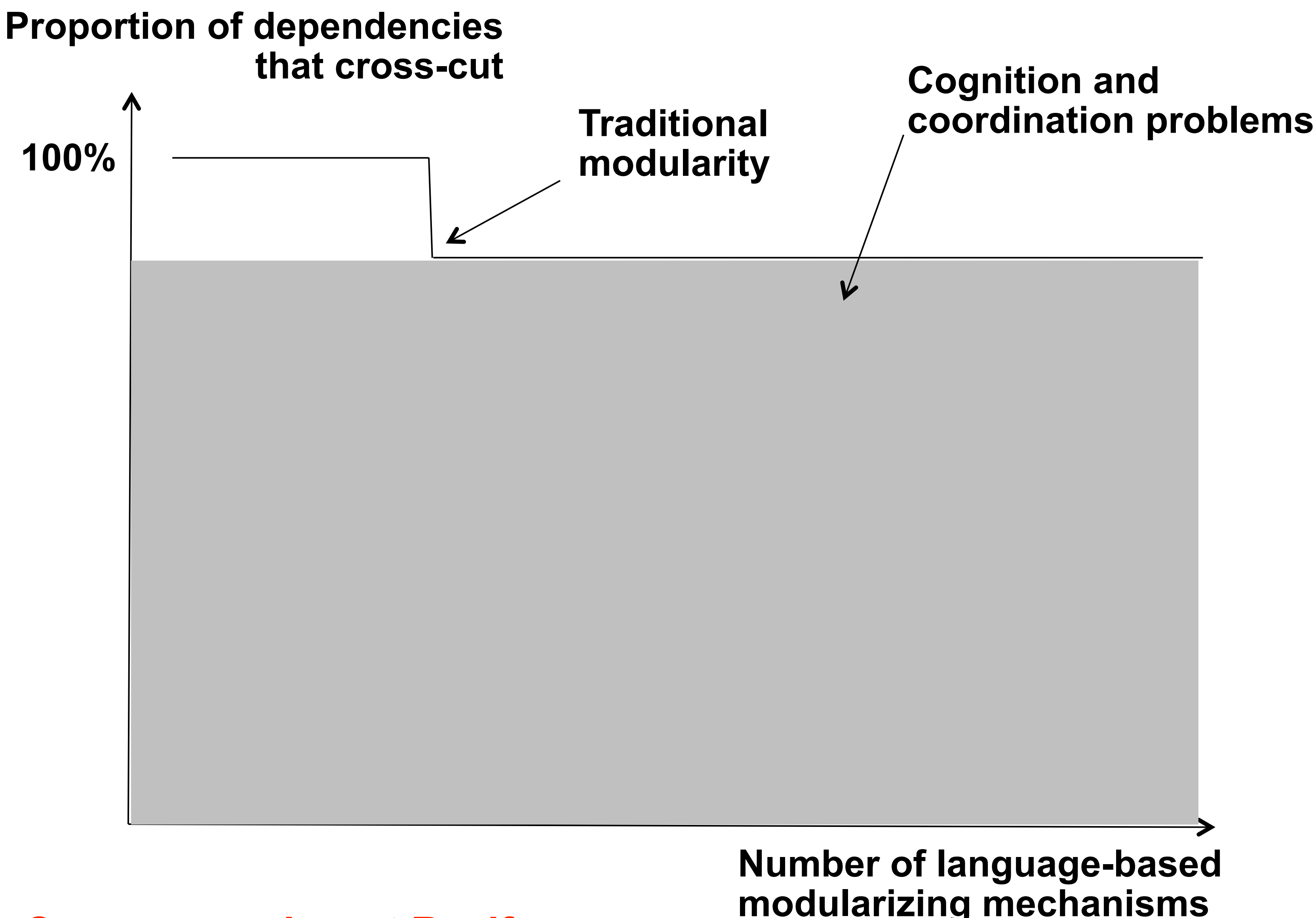

Cognition and
coordination problems
Proportion of dependencies
that cross-cut
100%
Number of language-based
modularizing mechanisms


Proportion of dependencies
that cross-cut
100%
Traditional
modularity
Cognition and
coordination problems
Number of language-based
modularizing mechanisms



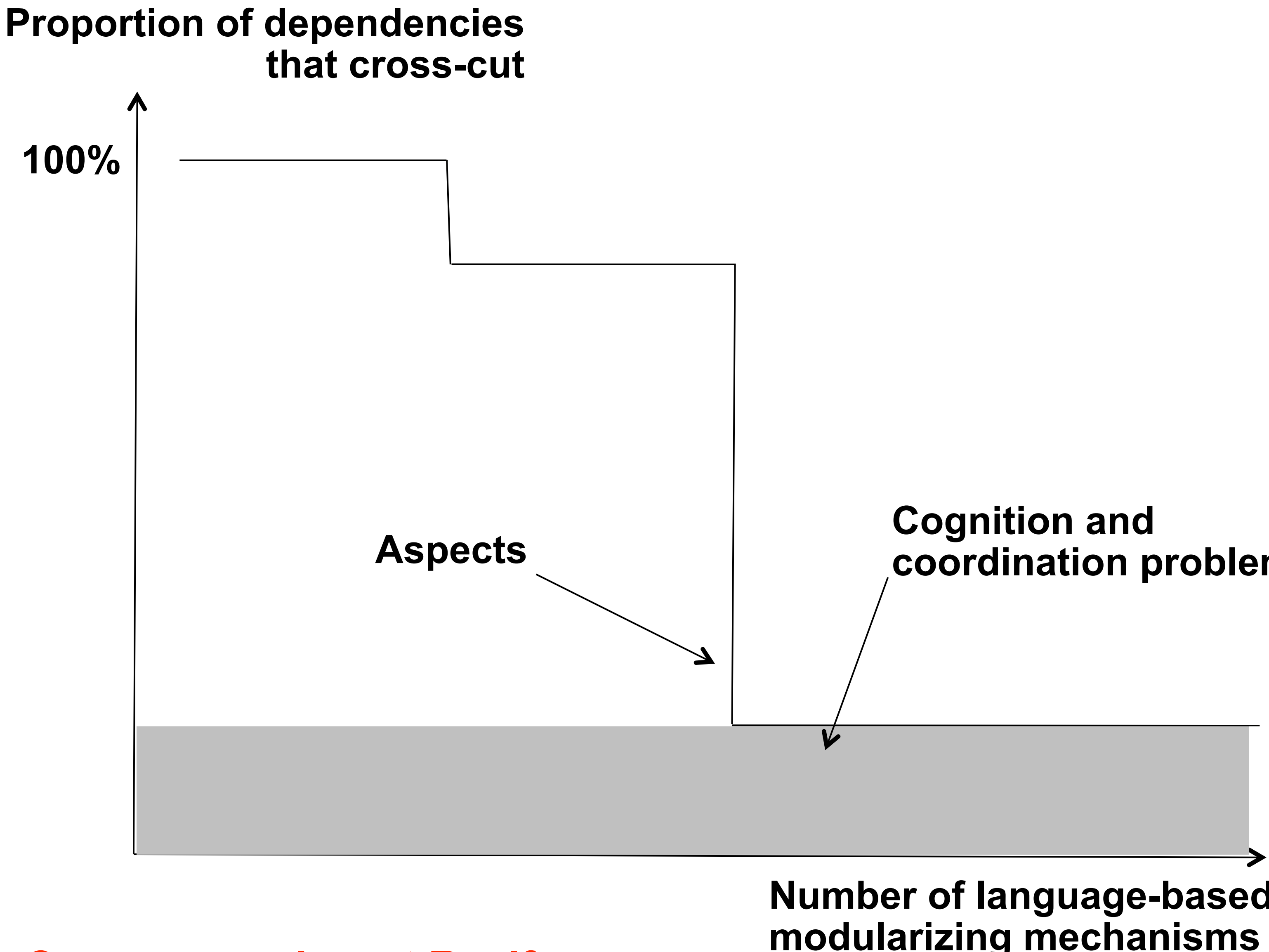
Carnegie Mellon
School of Computer Science


Proportion of dependencies
that cross-cut
100%
Cognition and
coordination problems
???
Number of language-based
modularizing mechanisms

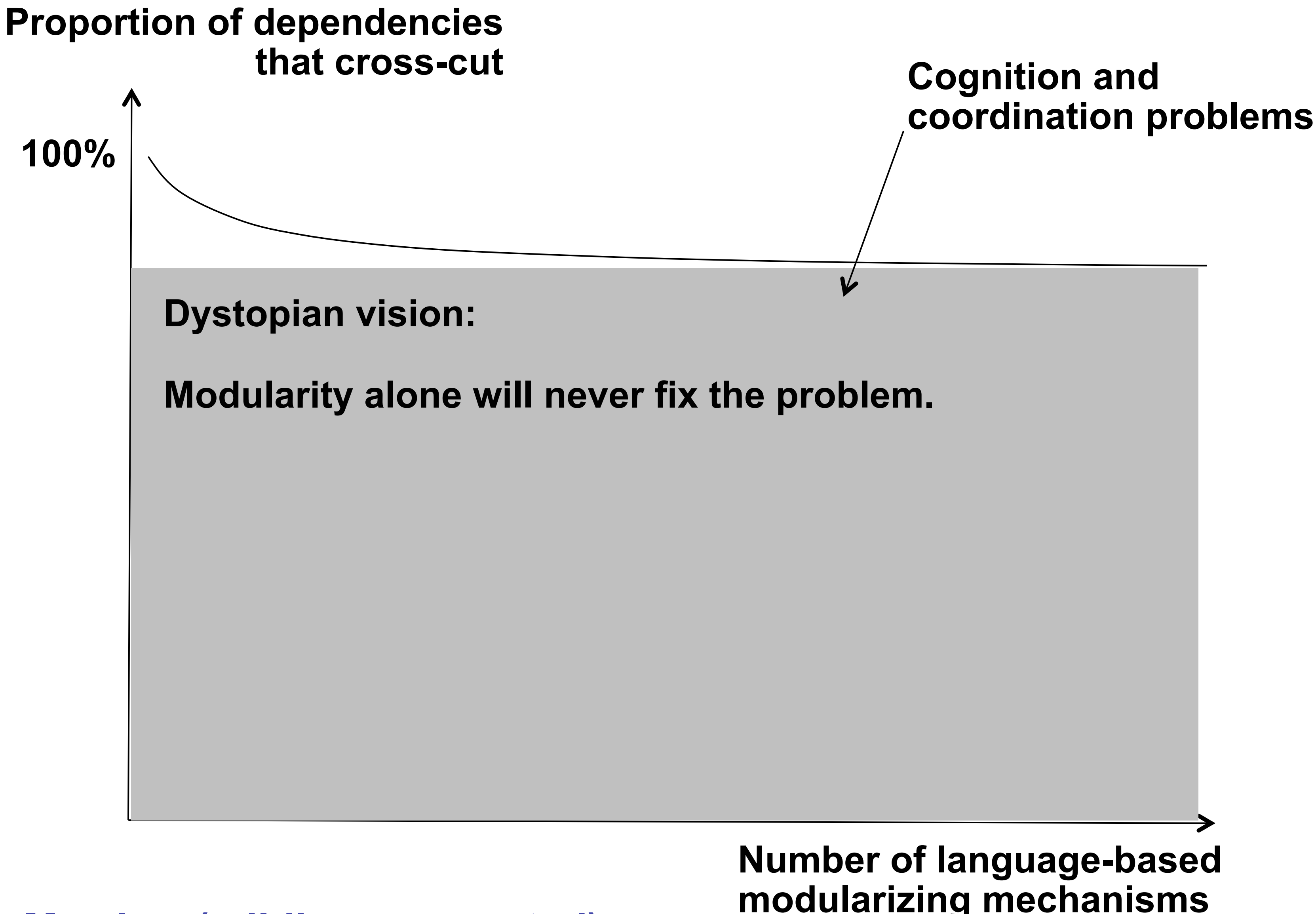


Consensus view at Recife

Carnegie Mellon
School of Computer Science



Consensus view at Recife



My view (mildly exaggerated)

Approaching the Gray Area . . .

- Organizational design, work assignment, and tools set up to bring the right dependencies to the attention of the right people so they can act appropriately

Two Examples . . .

- Organizational design and work assignment
  – Lessons from feature-driven development
- Using information from the environment
  – Learning from human activity

Feature-Driven Development

- Unit of functionality in end-user terms
- Feature is the unit of development managed by a project
- Features tend to cut across traditional software entities
- Work often overseen by “feature manager”
- Developers associated with component, assigned to work on particular features

The Study

- Setting
  - Software for automotive navigation system
  - 1195 features
  - 32 months of activity
  - 179 engineers in 13 teams
  - 1.5 M LOC, 6789 source files, 107 architectural components
  - Organization had 5 years of prior experience with feature-driven development
- Architects prepare feature development specification

What Causes Integration Failure?

	Model I	Model II	Model III	Model IV
Time	0.992*	0.990*	0.990*	0.989*
Average Component Experience (log)	0.487*	0.984+	0.741+	0.754
Changed LOCs		1.021	1.089	1.063
Concentration of Changed LOCs		1.045	1.028	1.036
Number of Dependencies (log)		1.107*	1.091*	1.091*
Concentration of Number of Dependencies		1.032**	1.046**	1.078**
Number of Groups			1.101*	1.051*
GSD			13.924**	14.964**
Feature Owner Belongs to Highly Changed Component			0.789	0.396
Feature Owner Belongs to Highly Coupled Component			0.839**	0.819**
Concentration of Changed LOCs X *F. Owner Belongs to Highly Changed Component*				1.032
Concentration of Number of Dependencies X *F. Owner Belongs to Highly Coupled Comp.*				0.977**
GSD X *Feature Owner Belongs to Highly Changed Component*				3.736
GSD X *Feature Owner Belongs to Highly Coupled Component*				0.926
Deviance of the Model	755.2	639.0	458.4	412.2
Deviance Explained	11.7%	25.3%	46.4%	51.8%

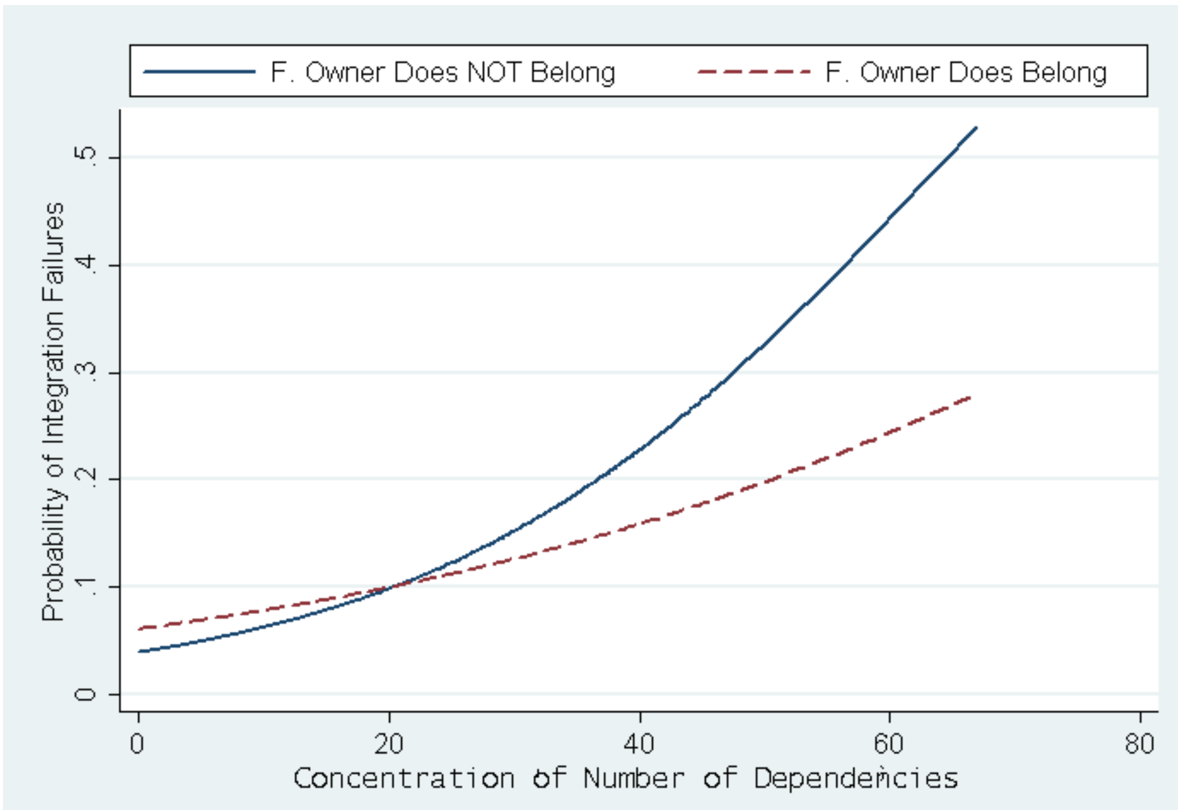
(+ p < 0.1; * p < 0.05; ** p < 0.01)

Odds Ratios from Regression Assessing Factors Driving Feature Integration Failures

From Cataldo, M. & Herbsleb, J.D. (2011). Factors Leading to Integration Failures in Global Feature-Oriented Development: An Empirical Analysis. *Proceedings, International Conference on Software Engineering* (to appear).

Carnegie Mellon

Ownership Matters!



From Cataldo, M. & Herbsleb, J.D. (2011). Factors Leading to Integration Failures in Global Feature-Oriented Development: An Empirical Analysis. *Proceedings, International Conference on Software Engineering* (to appear).

isr institute for SOFTWARE RESEARCH

Carnegie Mellon

Destructive Feature Interaction

	Model I	Model II	Model II
Time	0.981**	0.971**	0.964*
Failures in the Past 5 Weeks	2.127**	1.125*	1.011*
Changed LOCs	1.371**	1.201**	1.203**
Average Component Experience (log)	0.837+	0.997	0.908
Number of Groups	3.006**	4.037**	6.345**
Overlap Among Groups	0.943**	0.919**	0.901**
Same Feature Owner	0.876**	0.871**	0.852**
GSD	4.501**	2.509**	4.895**
Number of Cross-Feature Dependencies (log)		2.911**	4.938**
Number of Groups X Number of Cross-Feature Dependencies			0.607
GSD X Number of Cross-Feature Dependencies			0.799**
Deviance of the Model	12873.9	9413.1	8043.1
Deviance Explained	33.4%	51.3%	58.4%

(+ p < 0.1; * p < 0.05; ** p < 0.01)

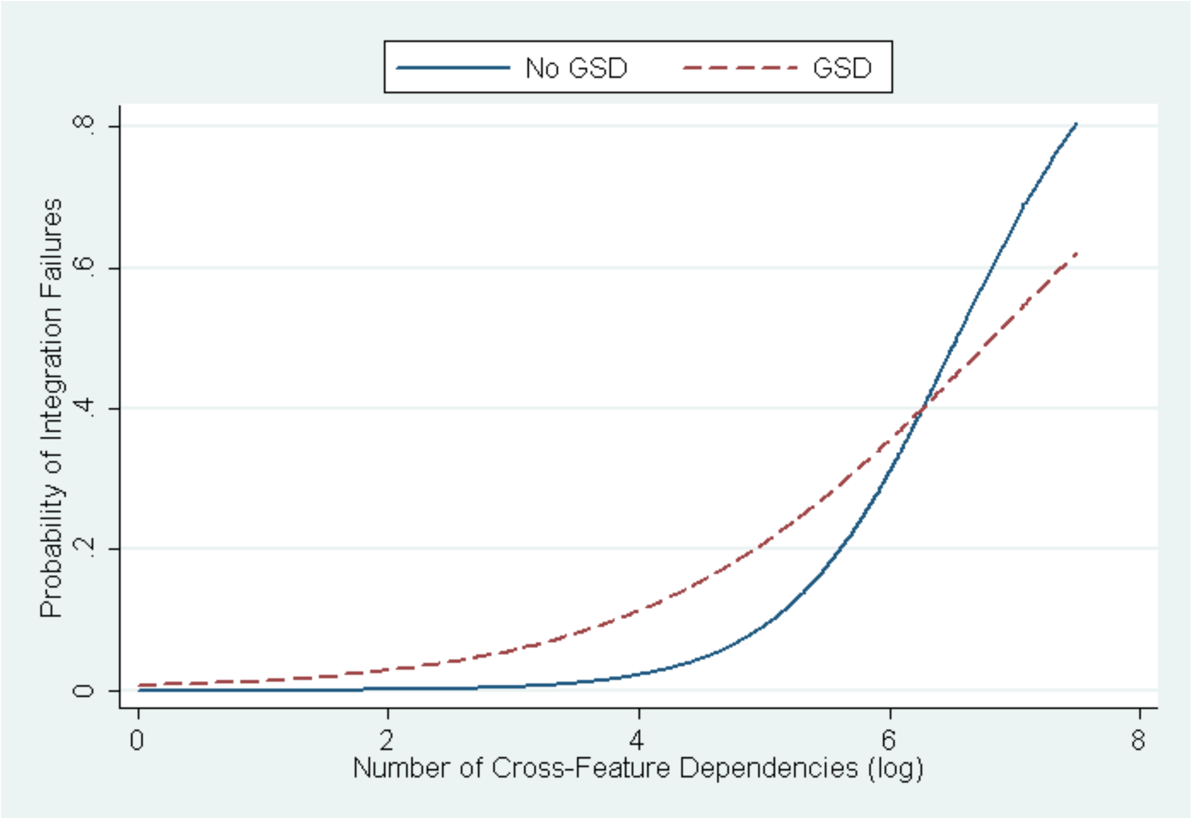
Odds Ratios from Regression Assessing the Impact of Cross-Feature Interactions on Integration Failures

From Cataldo, M. & Herbsleb, J.D. (2011). Factors Leading to Integration Failures in Global Feature-Oriented Development: An Empirical Analysis. *Proceedings, International Conference on Software Engineering* (to appear).

institute for SOFTWARE RESEARCH

Carnegie Mellon

Co-location Doesn't Scale



From Cataldo, M. & Herbsleb, J.D. (2011). Factors Leading to Integration Failures in Global Feature-Oriented Development: An Empirical Analysis. *Proceedings, International Conference on Software Engineering* (to appear).

institute for SOFTWARE RESEARCH

Broader Lessons

- Organizational arrangements matter!
- Effects can be quite large
- Effects often are not commonsensical

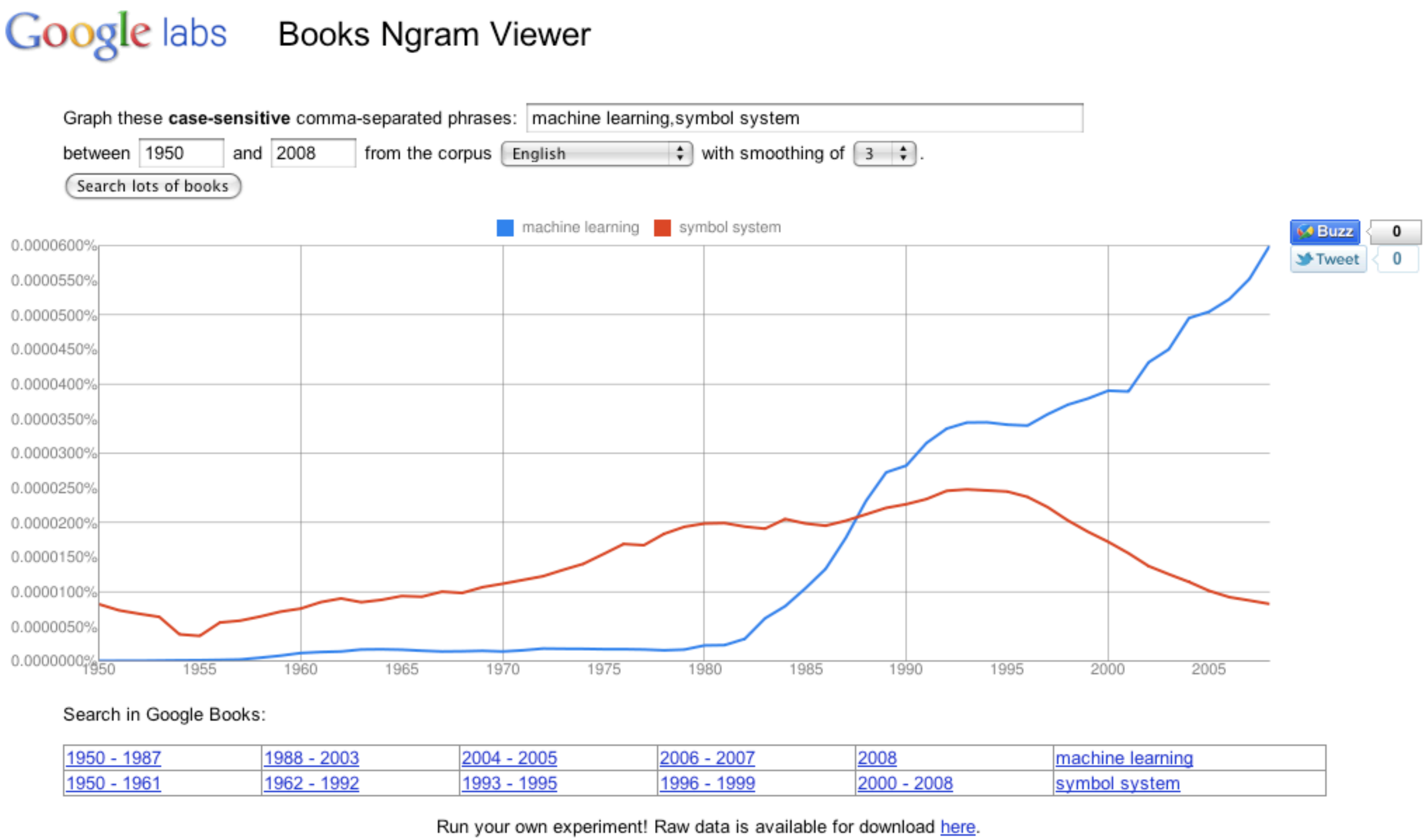
Carnegie Mellon

Inferring Dependencies from Traces of Human Activity

- Prior work
  - Use files changed together as measure of dependencies
  - Can generate a measure of coordination requirements
  - Validated in a number of settings
- Can we generalize from “files changed together” to “entities discussed together”?

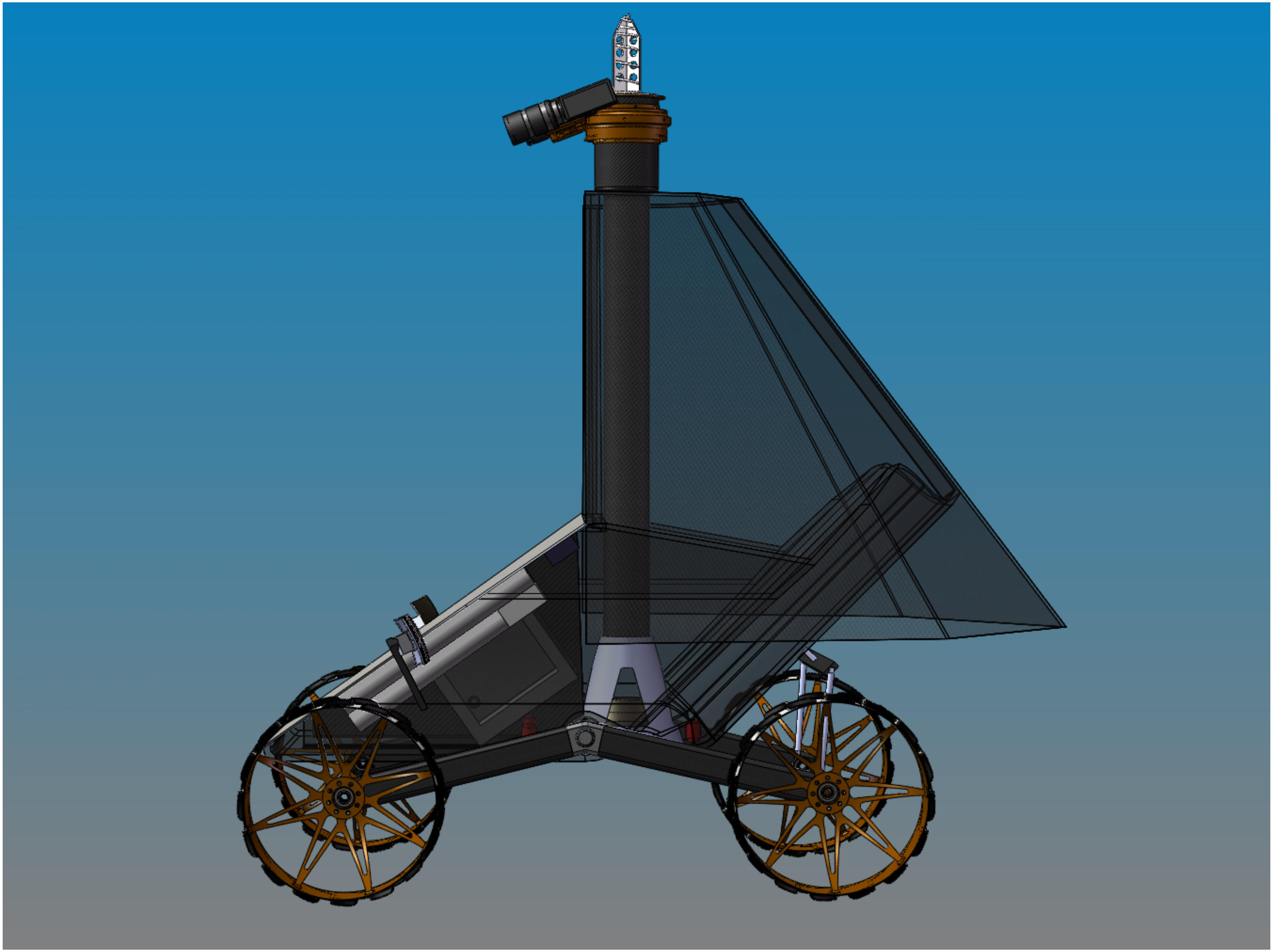
institute for SOFTWARE RESEARCH

A Brief Digression/Analogy



Text Analysis: Field Robotics

- Project
  - Lunar X Prize competition



Text Analysis: Field Robotics

- Project
  - Lunar X Prize competition
- No automatically collected version or change data
- Constantly shifting component boundaries and interfaces
- Can we use text analysis to derive dependencies?

Steps

- Collected data
  - 25 all-hands meetings
  - About 10,000 words each
- Developed code book
  - 6 field robotics articles

Carnegie Mellon

Code Book

Component:	Brief description:
Communications	Communications external to the robot, for control or mission, including operator interface.
External Relations	Acquiring external resources (incl. funding, parts, & purchasing), publicity, investor & media relations.
Internal Relations	Project/program management, HR, task assignments, training, collaboration tools, clarifying norms & expectations.
Mobility Effectors / Actuators	Effectors and actuators that propel the entire robot: e.g. tracks, wheels, shocks, & motors with associated firmware.
Mission Specific Effectors / Actuators	All other motors, gears, & moving parts that don't move the robot as a whole, e. g. camera mast rotation motor.
Perception software / computing	Software, and any dedicated hardware, for: terrain mapping, environmental modeling, and/or object detection. Camera/lens zoom, shutter, and focus control software.
Planning software / computing	Mission task planning, including the overall mission plan and computing resources for semi-autonomous execution.
Power	Includes batteries, solar cells, switches, power cables & controls.
Sensors	Camera; thermal, ultrasonic, tactile, radar/sonar range sensors; Inertial Measurement Unit, GPS, & any wiring or processing going from sensors to controls.
Shared / general computing	Includes general purpose processors / onboard computers (e. g. avionics box). Abbreviated "gpp."
Structure	Chassis, fasteners (e.g. Frangibolt, weld joints), radiator, payload, paints, reflectors.

isr institute for SOFTWARE RESEARCH

Steps

- Collected data
  - 25 all-hands meetings
  - About 10,000 words each
- Developed code book
  - 6 field robotics articles
- Manual coding of decision discussions
  - Tested inter-rater reliability
    - QAP correlations .80

Carnegie Mellon

Text Pre-Processing

Step	Description
1	Removed contractions (e. g. changing "what's" to "what is").
2	Applied a Krovetz (dictionary-based) stemmer to covert terms into morphemes
3	Removed common English terms (e. g. 'the'), replacing them with placeholders ('xxx').
4	Removed punctuation.
5	Turned meaningful bigrams into unigrams (e.g. 'solar cells' became 'solar_cells').

isr institute for SOFTWARE RESEARCH

Carnegie Mellon

Steps

- Collected data
  - 25 all-hands meetings
  - About 10,000 words each
- Developed code book
  - 6 field robotics articles
- Manual coding of decision discussions
  - Tested inter-rater reliability
    - QAP correlations .80
- Created thesaurus

institute for SOFTWARE RESEARCH

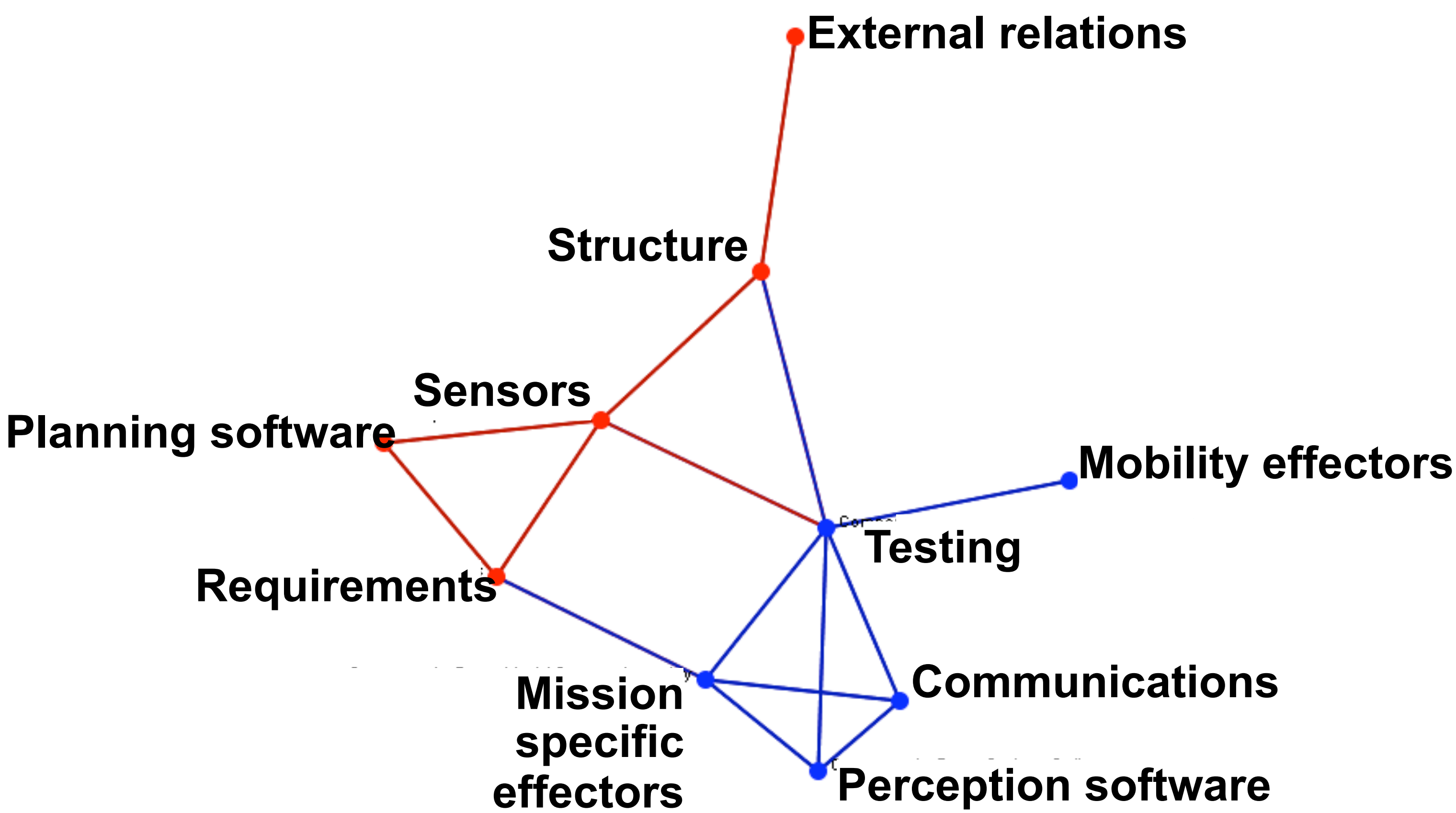
Carnegie Mellon

Link Identification

- Co-occurrence of terms
- Human coding: same decision
- Selected sliding window size
  - Size 15 had best agreement with hand coding
  - Threshold established
- QAP correlations comparable to human-human agreement (~.8)
- Sets of links based on topics

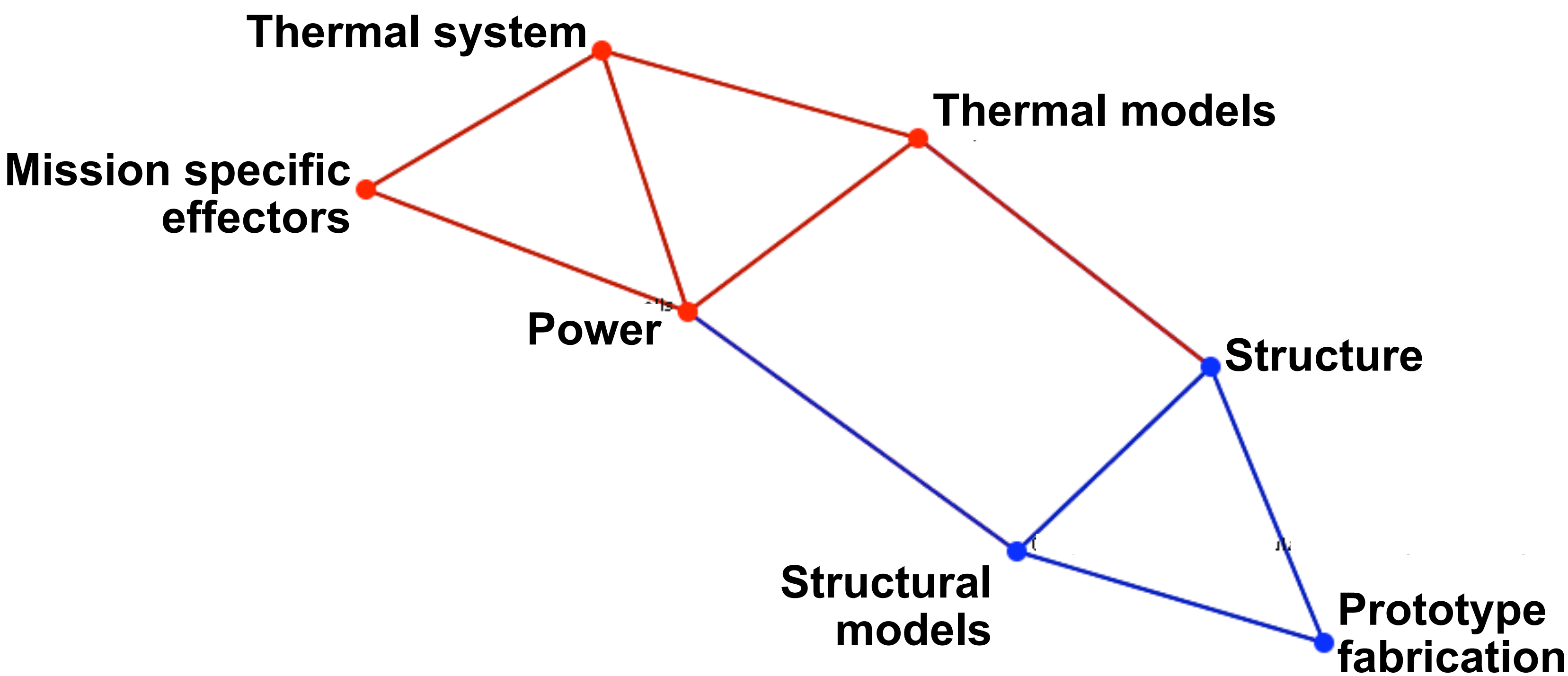
institute for SOFTWARE RESEARCH

Optics


External relations
Structure
Sensors
Planning software
Mobility effectors
Testing
Requirements
Communications
Mission specific effectors
Perception software

Carnegie Mellon

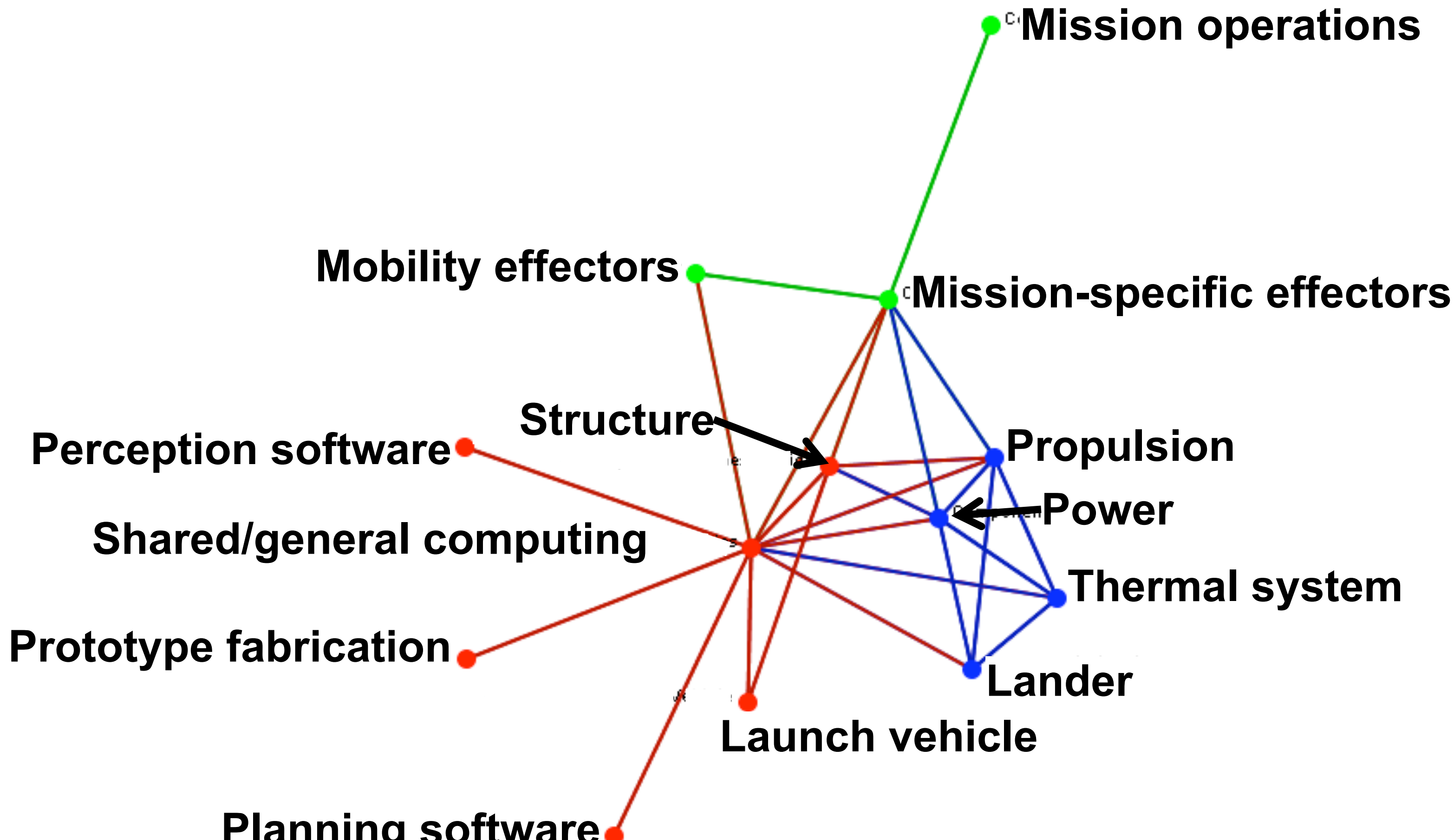
Thermal



isr institute for SOFTWARE RESEARCH

Carnegie Mellon

Avionics



institute for SOFTWARE RESEARCH

Carnegie Mellon

Concluding Vision

- The gray area – work that cross-cuts language constructs – is here to stay
- Use organizational tactics
- Use computations over artifacts generated by development activities
- Explore new data sources, including documents and conversation
    - Activities reveal knowledge
    - Analysis can often make it actionable

institute for SOFTWARE RESEARCH