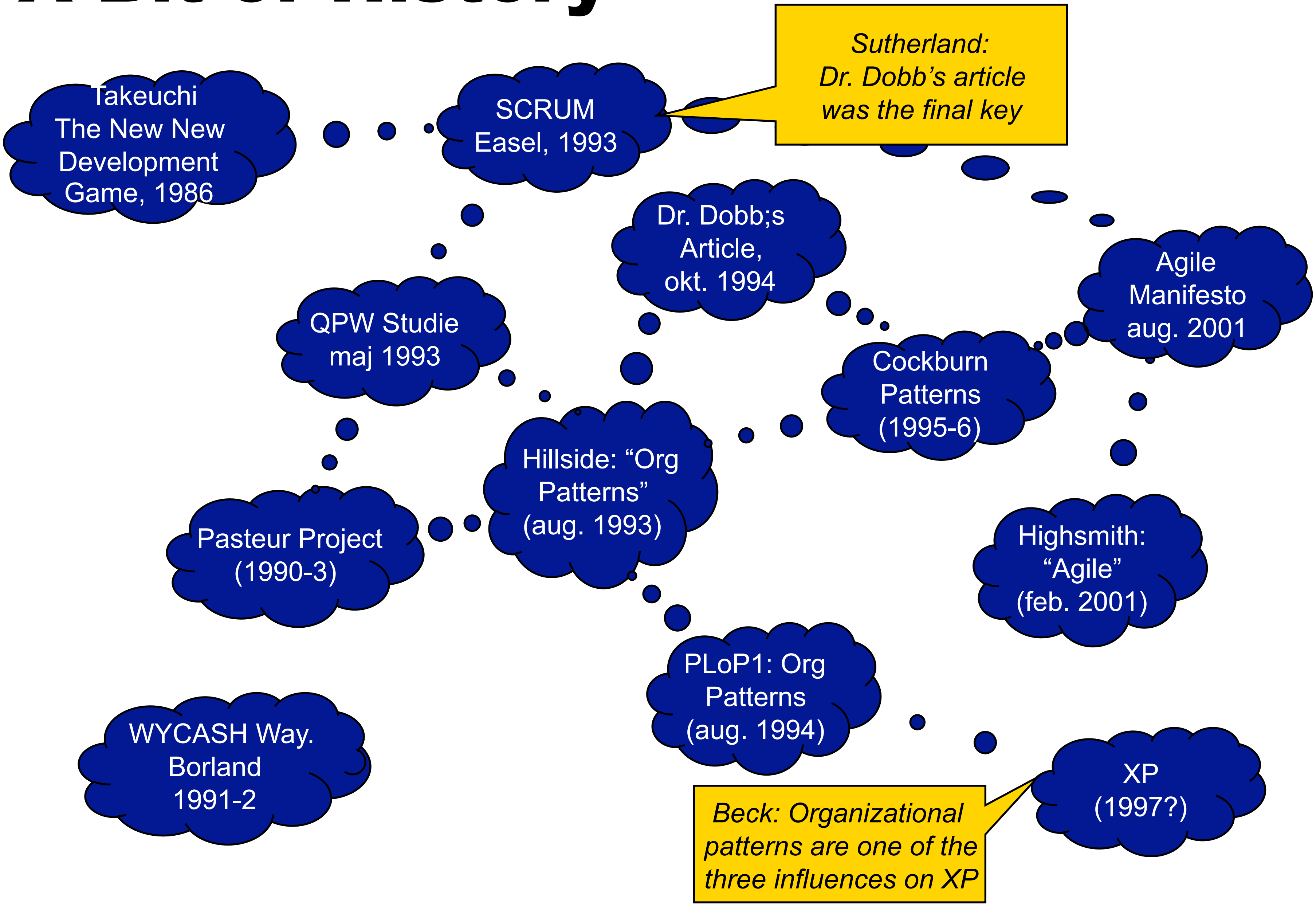
Organizational Patterns and Agility

James O. Coplien

Nordija A/S

A Bit of History



Takeuchi
The New New
Development
Game, 1986

SCRUM
Easel, 1993

*Sutherland:
Dr. Dobb’s article
was the final key*

Dr. Dobb;s
Article,
okt. 1994

QPW Studie
maj 1993

Agile
Manifesto
aug. 2001

Cockburn
Patterns
(1995-6)

Hillside: “Org
Patterns”
(aug. 1993)

Pasteur Project
(1990-3)

Highsmith:
“Agile”
(feb. 2001)

PLoP1: Org
Patterns
(aug. 1994)

WYCASH Way.
Borland
1991-2

XP
(1997?)

*Beck: Organizational
patterns are one of the
three influences on XP*

Organizational Patterns: born out of a need for agility

- The ISO 9000 world of 1990 telecom
  - **Process heaven: linear shelf decimeters of process documentation: the “Online Methodology” (OLM)**
  - **“Highly compliant” organization…**
  - **… yet 80% of the work was being done under “documented waivers”**
  - **Employees outside earshot of ISO auditors would decry the OLM as a myth**
- Our conclusion: Process guidance didn’t work
  - **Premise: roles are more stable than process, and needn’t change**
  - **Focusing on roles and responsibilities allows the right things to happen**

Patterns: Our Tool of Empiricism and Learning Processes

- A solution to a problem in a context
- Architectural patterns ideas first published by Alexander in 1977
- Look at issues of system structure, not just parts
- Build on proven practice, not just promising theories
- Have a central notion of the *fundamental process*:
  - **1. Find the weakest link**
  - **2. Fix it locally by adding local structure**
  - **3. If it is better, iterate. If not, undo it and try restructuring elsewhere**
- It's how organizations learn

More formally (for you academics)

- A pattern language L defines a geometry S characterized by a semi-group G
- A pattern $p \in G$ such that $p: S \rightarrow S$
- The language L constrains the order of composition $p_m \bullet p_n$ (can be formalized with a binary predicate on each function, reduces to I)
- There exists a weak group invariant over G (G is a similarity group)
- Each pattern p reduces the entropy of a problem vector space X

What are Organizational Patterns?

- Solutions to *organizational* problems in a context
- First appeared in the Alexander + software context at PLoP in 1994 (Coplien, Whitenack); received with some skepticism
- Now, a growing body of knowledge

Or, a construct from anthropology, Kroeber:

Universal patterns: transcend cultures
Systemic patterns: have a common root in an ancient culture
Total culture patterns: give a culture its identity

Patterns define culture

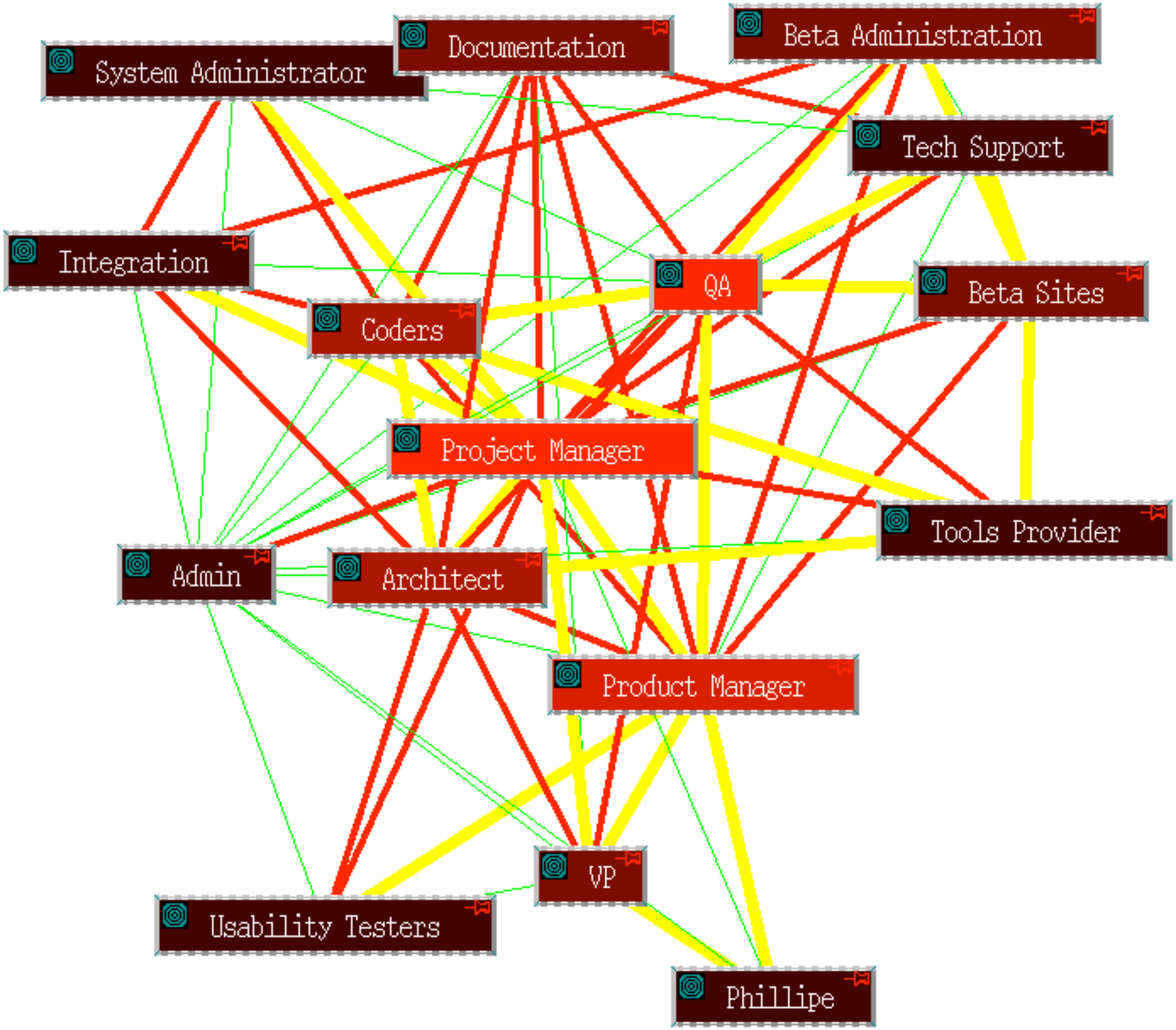
Mining the Patterns: Work-life Role-Play

- Identify project roles
- Study subjects play roles
- Development scenarios drive role-play
- Capture interaction & coupling on CRC cards
- Social Network Analysis Tools
  - **Organization Structure Visualization**
  - **Organizational Metrics**
- Capture Trends as Generative Patterns

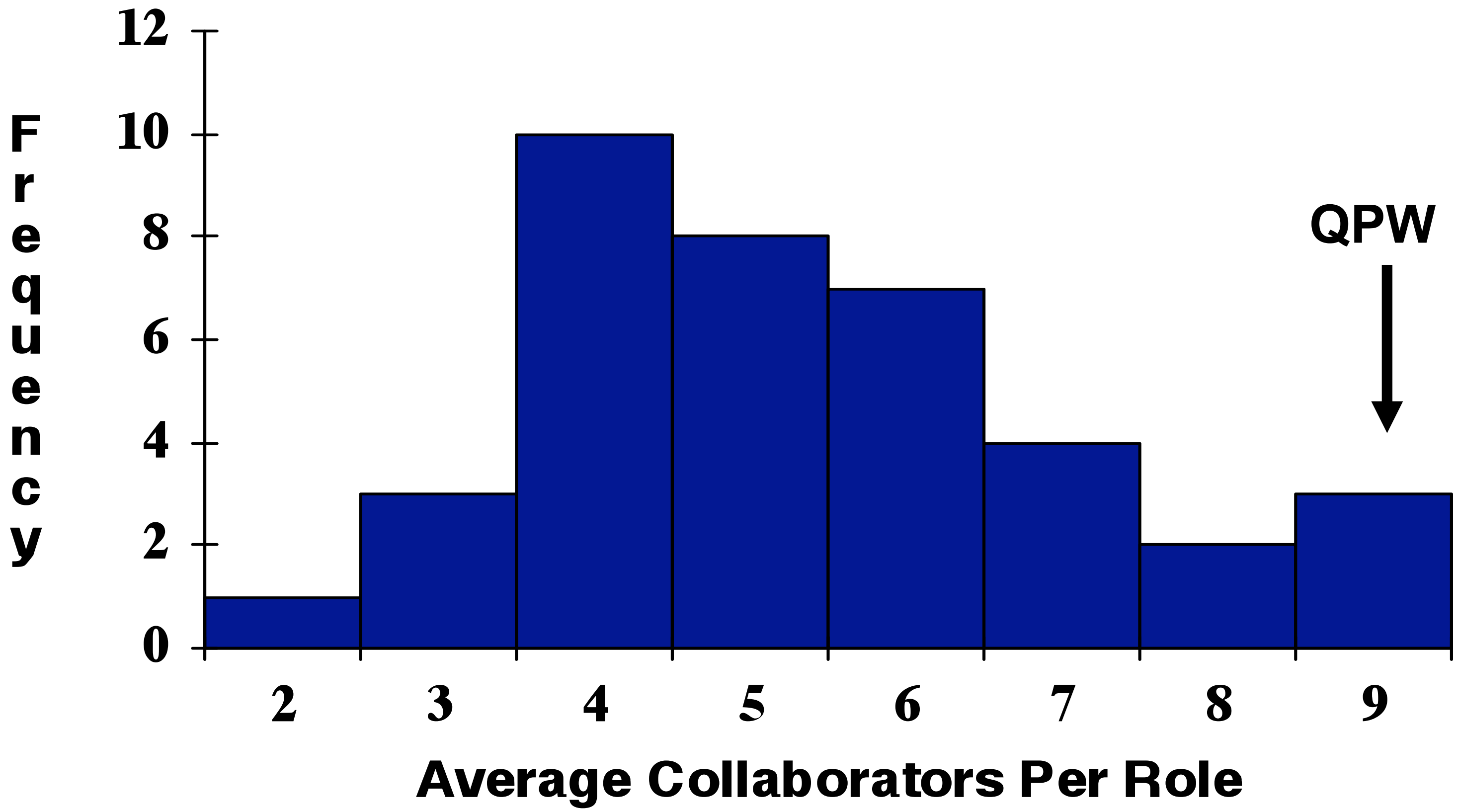
CRC Cards: Classes, Responsibilities, and Collaborators

Subsystem coord.

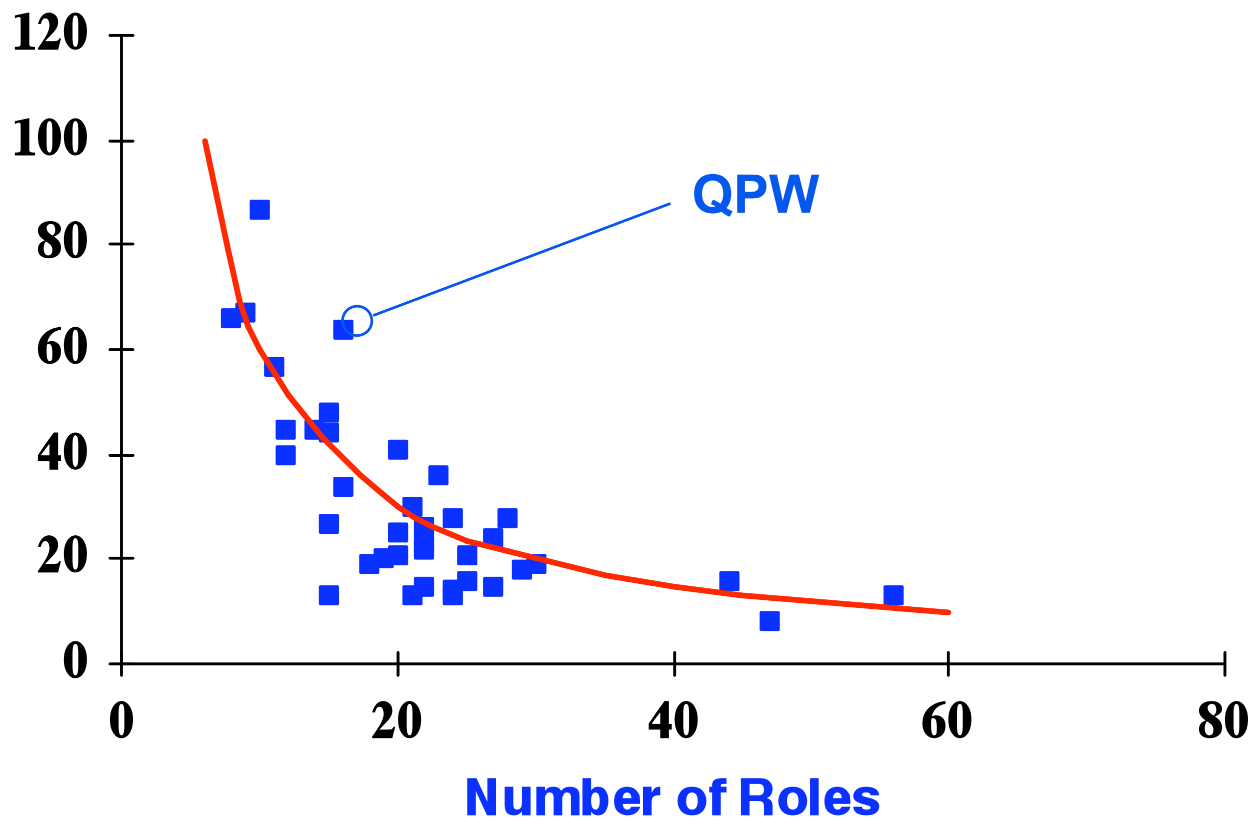
Validate MR lists	*Subsystem coord.*
Build products	*Change committee*
Administer ENVY	*Designers*
Resolve deps.	*System test*

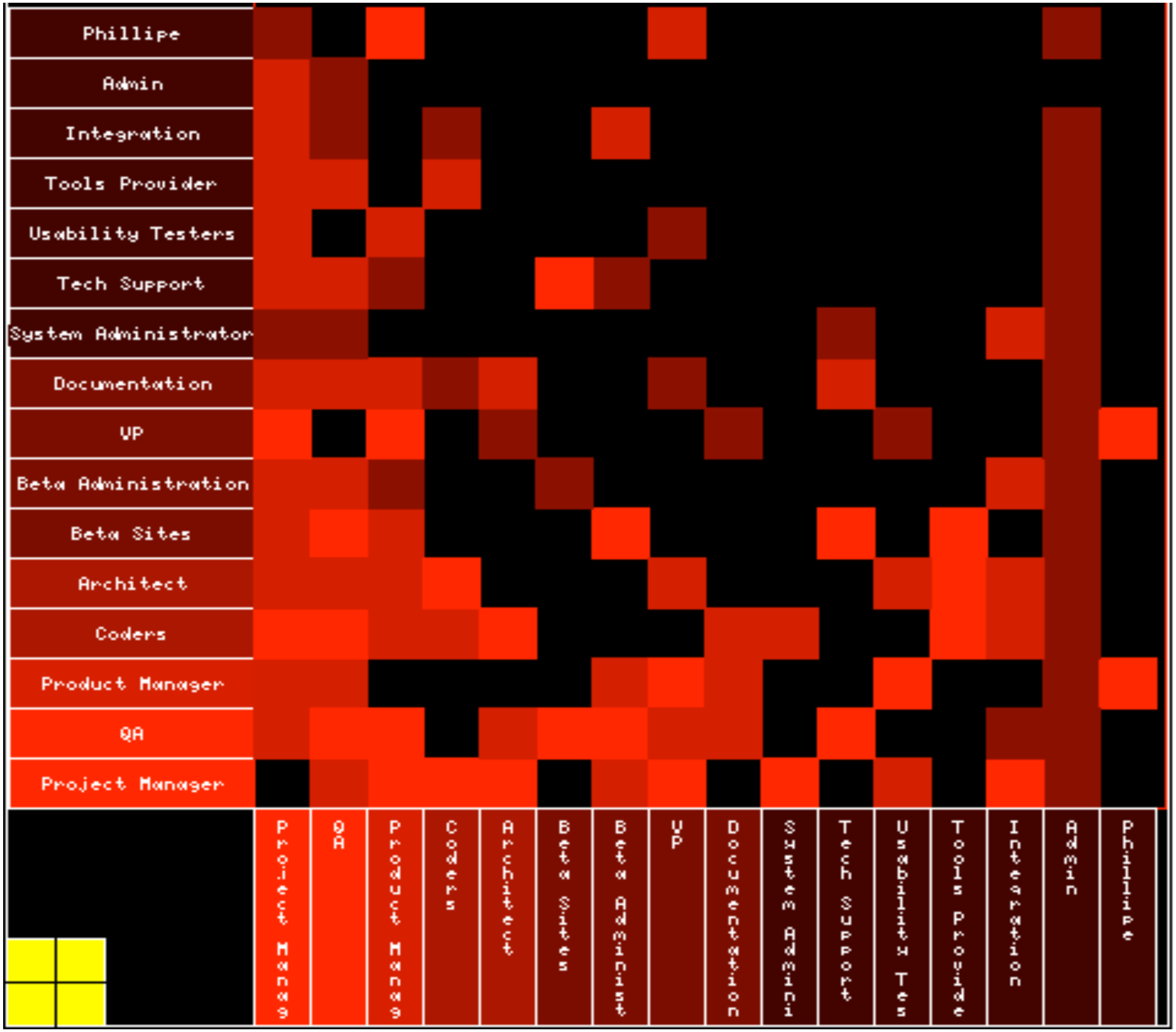

System Administrator
Documentation
Beta Administration
Tech Support
Integration
QA
Beta Sites
Coders
Project Manager
Tools Provider
Admin
Architect
Product Manager
VP
Usability Testers
Phillipe

Organization Metrics



Agility is about communication




Phillipe
Admin
Integration
Tools Provider
Usability Testers
Tech Support
System Administrator
Documentation
VP
Beta Administration
Beta Sites
Architect
Coders
Product Manager
QA
Project Manager
Project Manag
QA
Product Manag
Coders
Architect
Beta Sites
Beta Administ
VP
Documentation
System Admini
Tech Support
Usability Tes
Tools Provide
Integration
Admin
Phillipe

Distribute Work Evenly

…an organization is working to organize in a way that makes the environment as enjoyable as possible and which makes the most effective use of human resources.

* * *

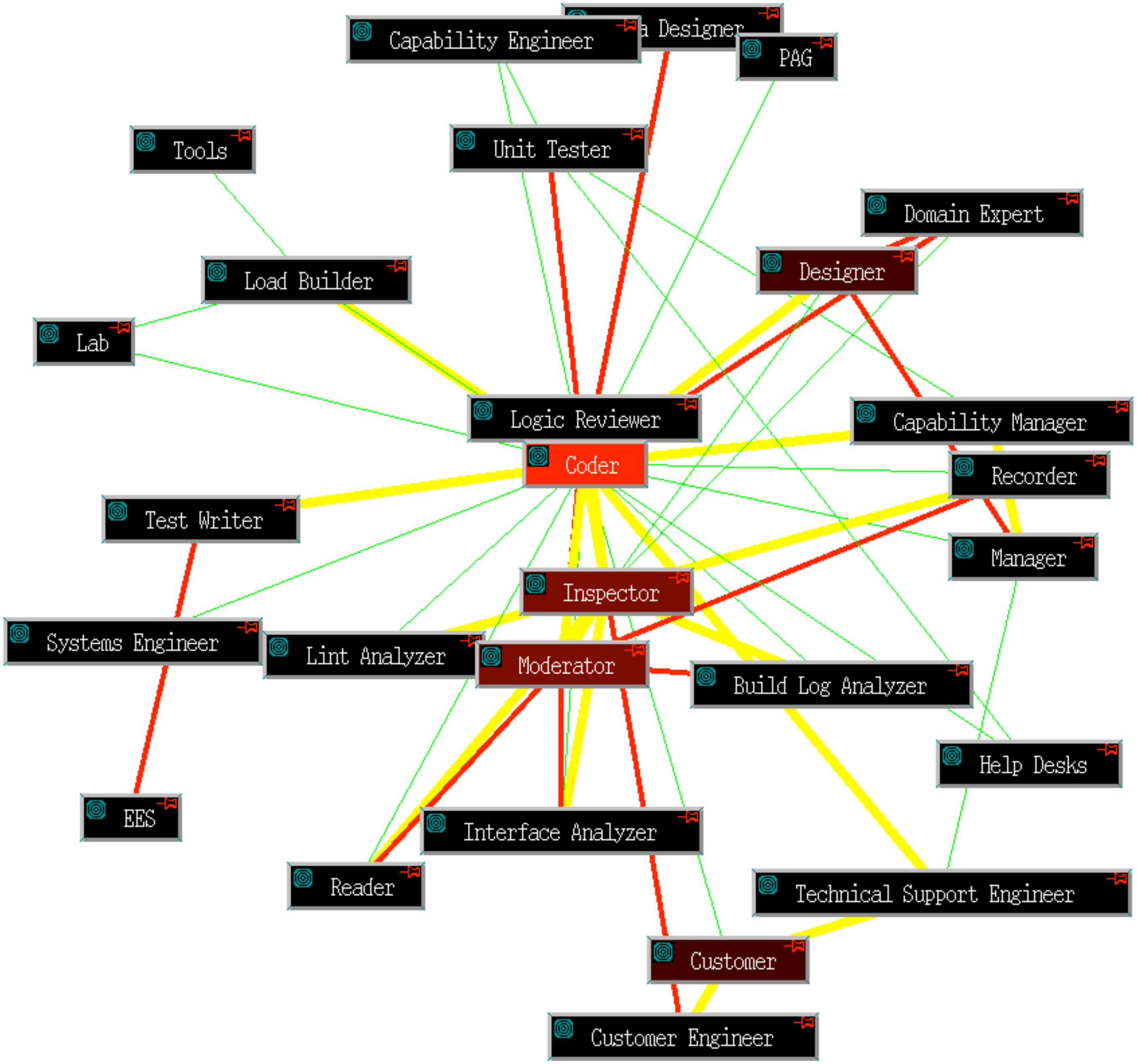
It is easy to depend on just a few people to carry most of the organization's burdens. Managers like this because it minimizes the number of interfaces they need to manage. And some employees strive to do all they can out of a misplaced feeling of monumental responsibility. In fact, we find that PRODUCER ROLES tend to have stronger communication networks than other support roles.

But if this unevenness continues, it is difficult for a heavily loaded role to sustain the communication networks necessary to healthy functioning of the enterprise as a whole. Resentment might build between employees who don't feel like they are central to the action. And the central people may easily burn out.

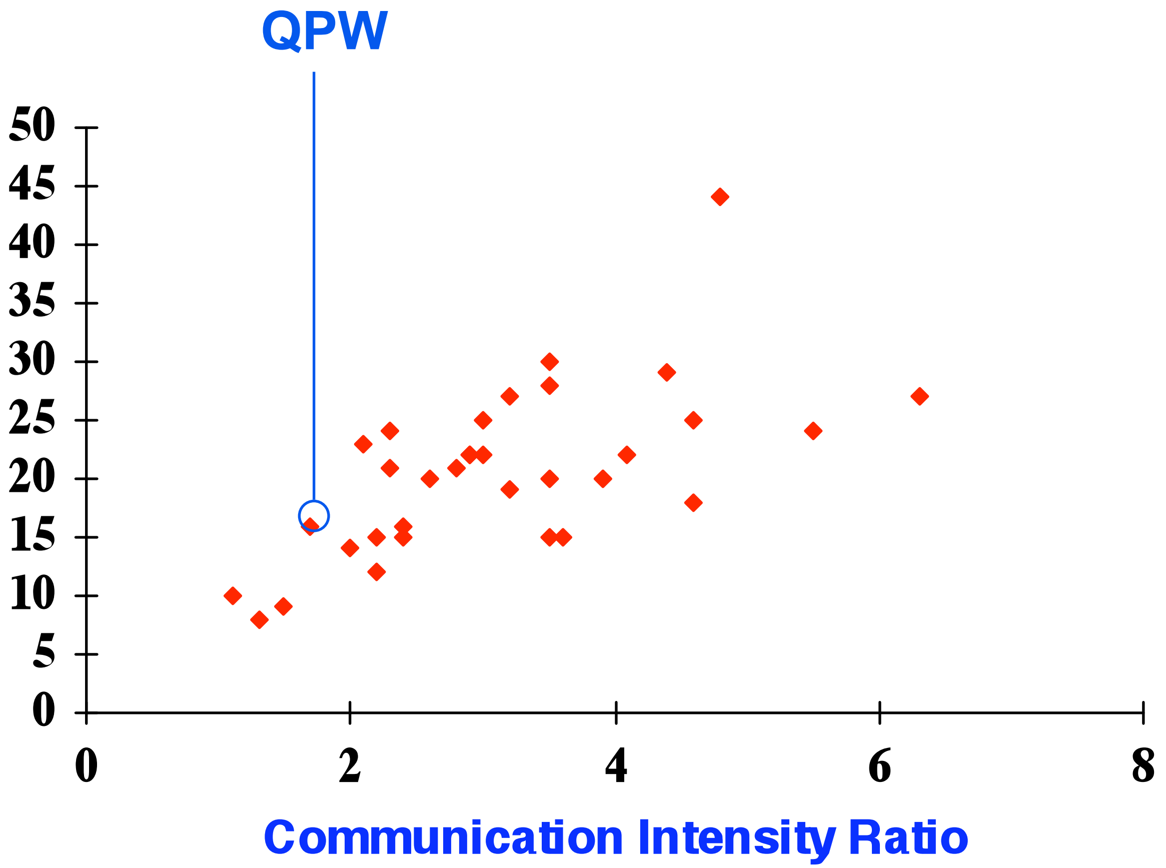
Define the communication intensity ratio as the ratio of the number of communication paths of the busiest role to the average number of communication paths per role. The organization has a problem if this ratio becomes too large.

Therefore: **Try to keep the communication intensity ratio to two or less**. (We have found that it isn't easy to get much below two.) The easiest way to do it is to have FEW ROLES. It also helps to identify the PRODUCER ROLES and eliminate any deadbeat roles. You can also identify all the communication to the most central role and see which are really necessary. There may be ways to "brute-force" eliminate some

of the communication, after you have identified it.


a Designer
Capability Engineer
PAG
Tools
Unit Tester
Domain Expert
Load Builder
Designer
Lab
Logic Reviewer
Capability Manager
Coder
Recorder
Test Writer
Manager
Inspector
Systems Engineer
Lint Analyzer
Moderator
Build Log Analyzer
Help Desks
EES
Interface Analyzer
Reader
Technical Support Engineer
Customer
Customer Engineer

Distribute Work Evenly



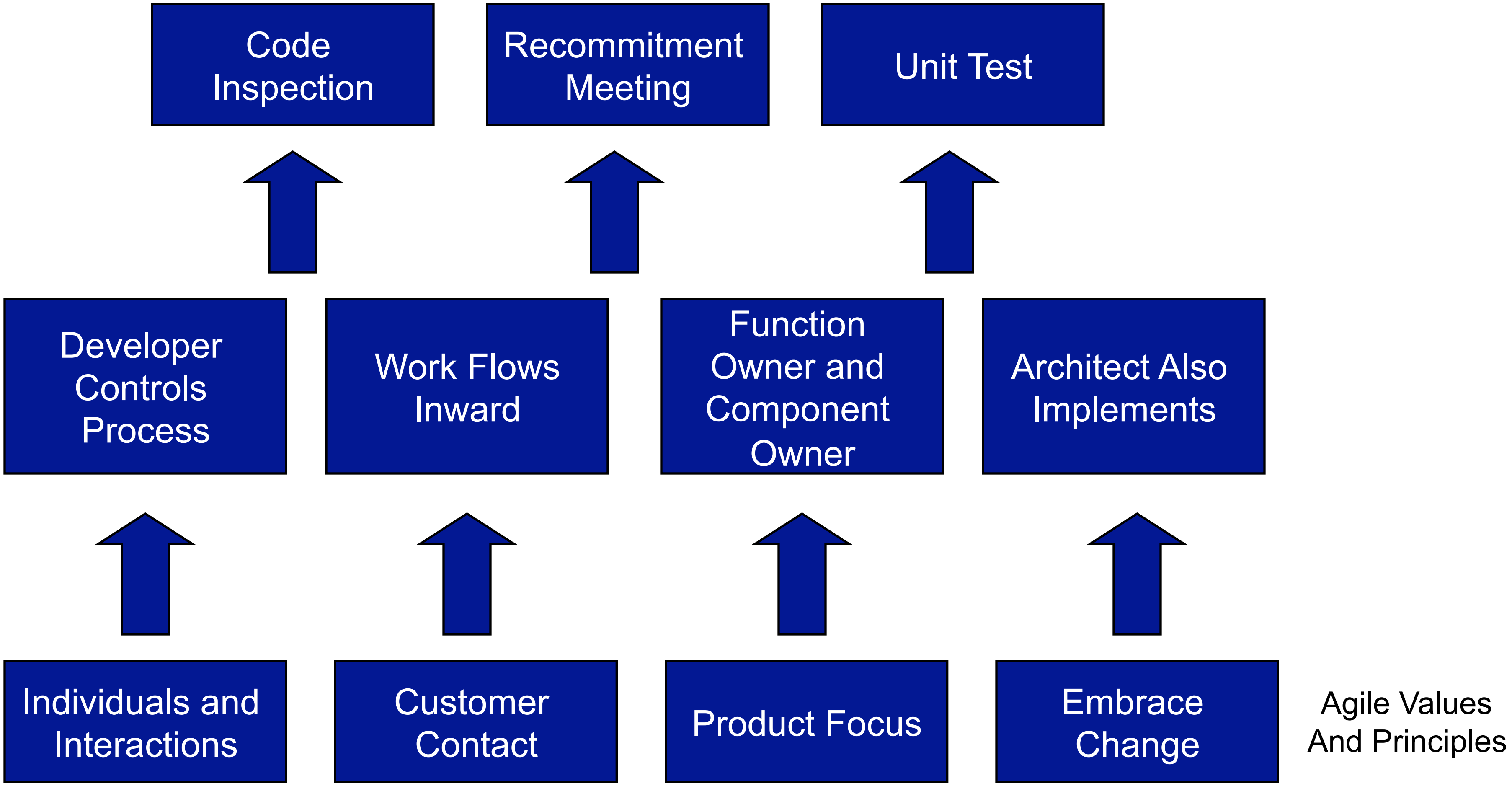
Patterns work together

- Like words in a language combine into sentences, you can combine patterns into an organization
- There are rules for putting patterns together
  - **For example, TEAM PER TASK provides context for PROGRAMMING IN PAIRS**
- However, there are many legal ways to put them together
  - **… because there are many kinds of organizations**
- Building the process itself should be agile

Systems Thinking and Patterns

- Where to attack the problem?
  - Processes: the ISO 9000 story: too superficial; comes from structure
  - Organizational structure: deeper, manageable; comes from values
  - Values: Relate to organizational identity; very difficult to elicit
- How to attack the problem?
  - Local adaptation and piecemeal growth: impossible to master-plan

Organizational patterns


Code Inspection
Recommitment Meeting
Unit Test
Developer Controls Process
Work Flows Inward
Function Owner and Component Owner
Architect Also Implements
Individuals and Interactions
Customer Contact
Product Focus
Embrace Change
Agile Values And Principles

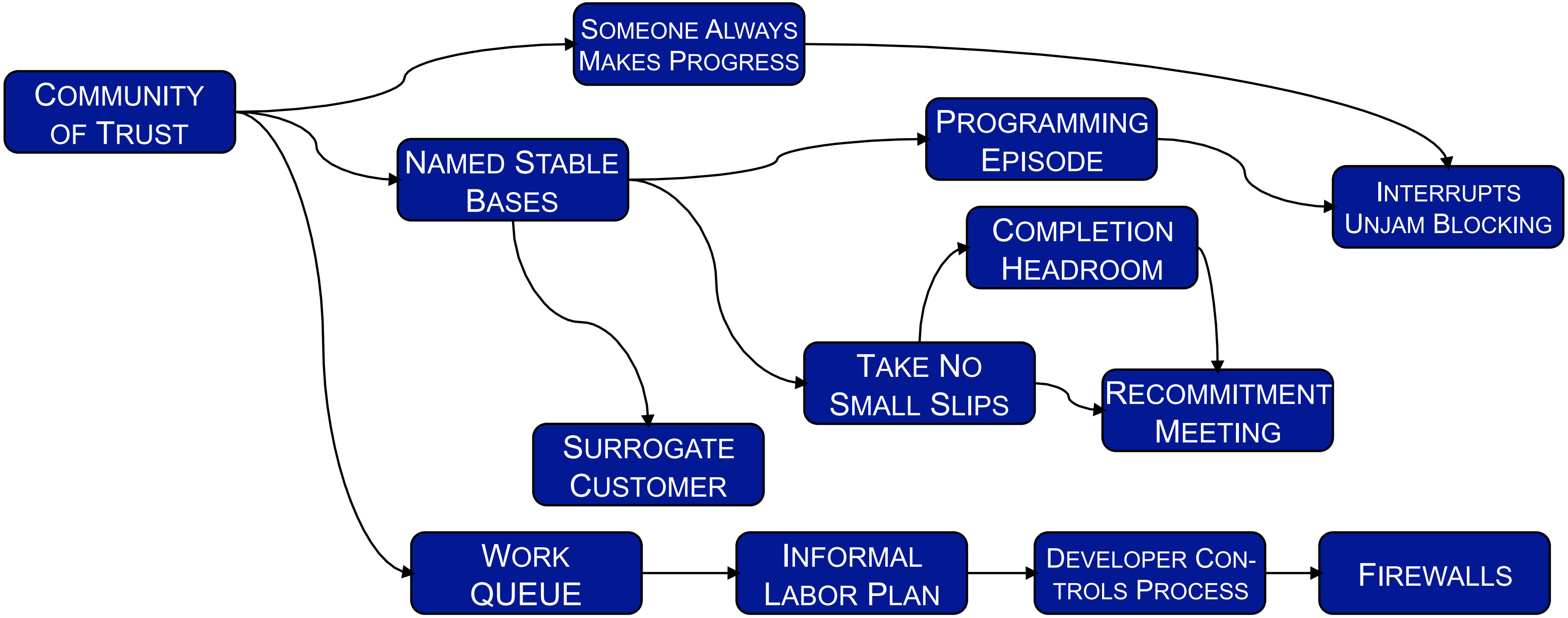
The Top Ten Org Patterns

- Unity of Purpose
- Engage Customers
- Domain Expertise in Roles
- Architect Controls Product
- Distribute Work Evenly
- Function Owner and Component owner
- Mercenary Analyst
- Architect Also Implements
- Firewalls
- Developer Controls Process

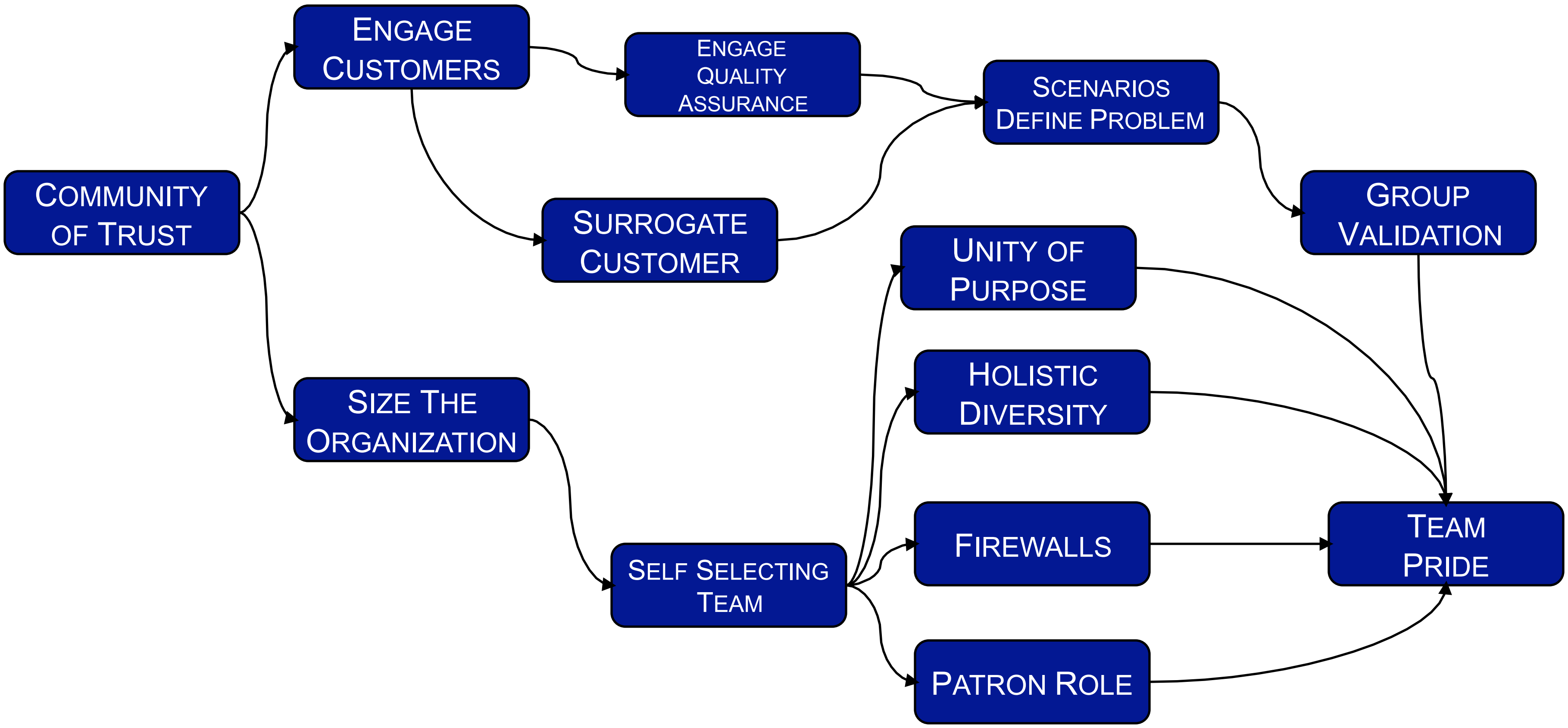
Scrum by the Org Patterns

Scrum in general	Few Roles, Producer Roles,
Sprint	Named Stable Bases, Take No Small Slips, Programming Episode, Work Queue, Informal Labor Plan, Developer Controls Process, Someone Always Makes Progress
Daily Scrum, Release Planning	Group Validation
Customer Demo	Engage Customers
Sprint Backlog	Someone Always Makes Progress
Burn-down Chart	Work Queue, Completion Headroom
Stop-the-Line	Recommitment Meeting, Take No Small Slips
Roles	Few Roles
Product Owner	Patron, Surrogate Customer
ScrumMaster	Firewalls
The Team	Self-Selecting Team, Producers in the Middle
DONE	Take no small slips, Named Stable Bases
Multi-disciplinary Team	Holistic Diversity
Chickens and Pigs	Firewalls, Producer Roles

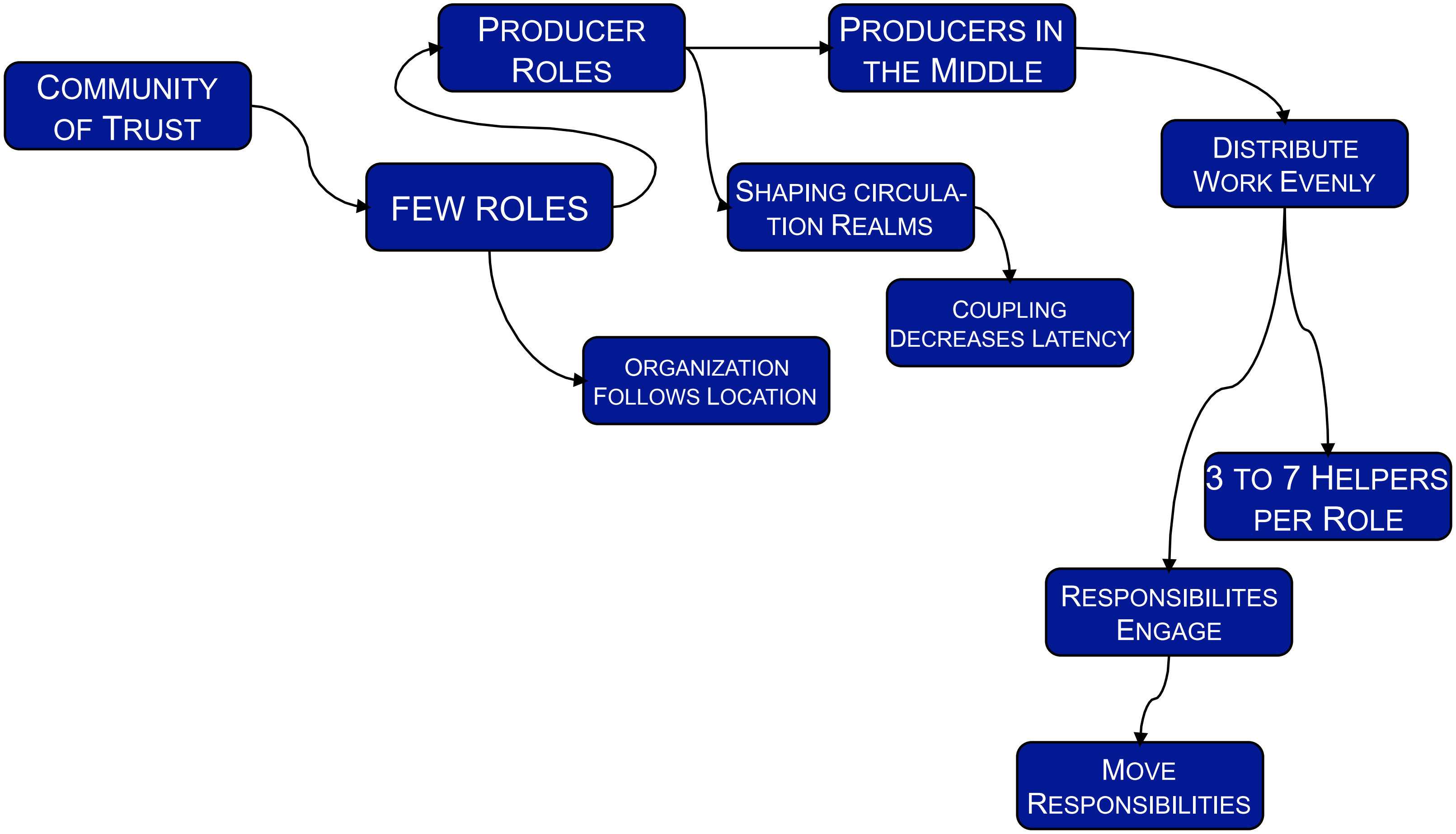
Scrum: Project Management Pattern Language



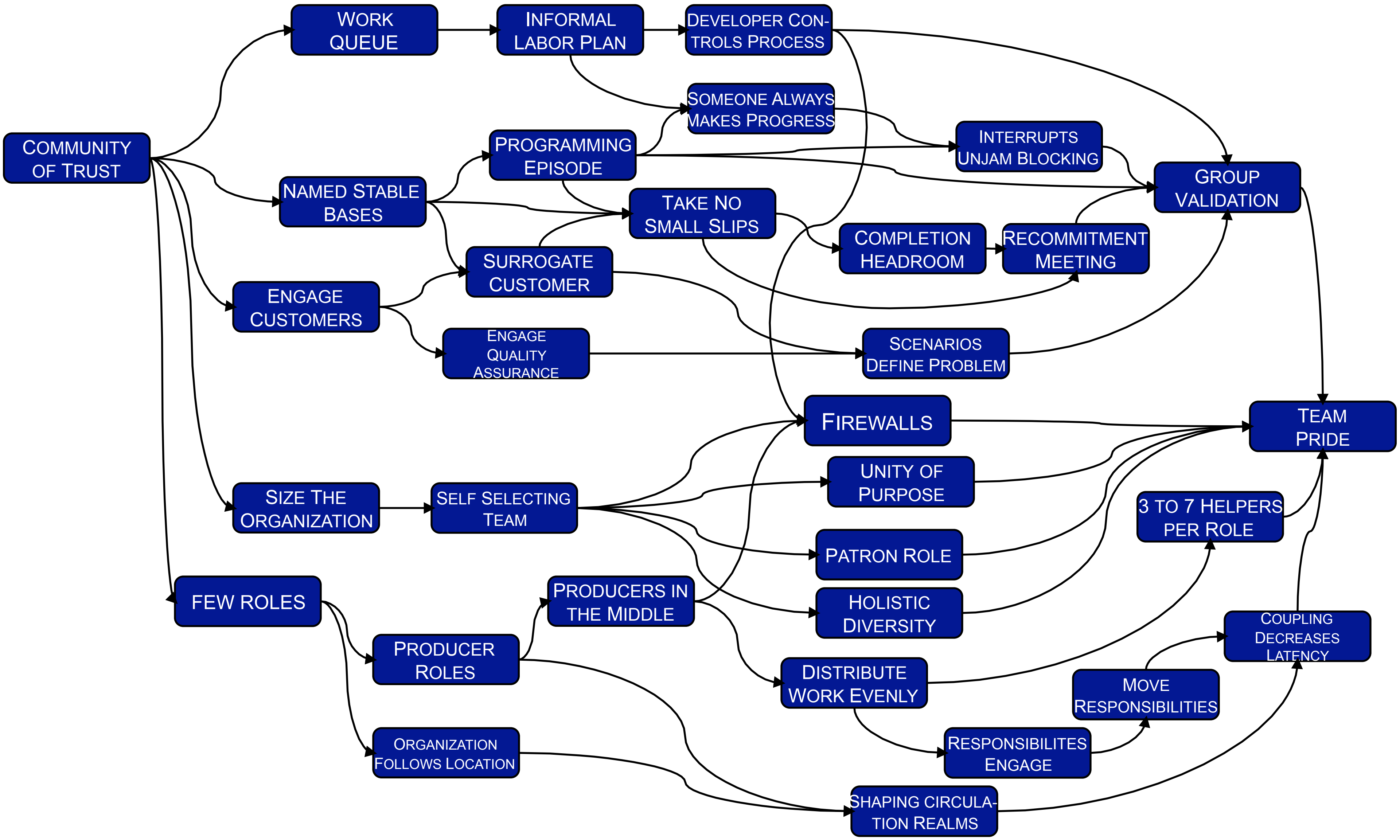
Scrum: Piecemeal Growth Pattern Language



Scrum: Organizational Construction Patterns



Scrum as Org Patterns



Conclusion

- Organizational Patterns capture Agile foundations
- Grounding in a decade of empirical research
- Be wary of trying Scrum before having "competencies" from the Org Patterns
- Patterns are an incremental, low-risk path to Agile adoption

Interesting On-Line Reading

- Sutherland, Jeff. SCRUM: Another way to think about scaling a project. 11 March 2003, on the web at Jeff Sutherland's SCRUM Log. On how the Organizational Patterns work is the foundation of SCRUM. http://jeffsutherland.org/scrum/2003_03_01_archive.html
- Schwaber, Ken. Scaling Agile Processes. In the Agile Project Management E-Mail Advisor, 3 April 2003. QPW as an example of scaling Agile processes. http://www.cutter.com/project/fulltext/advisor/2003/apm030403.html
- Coplien, James. Borland Software Craftsmanship: A New Look at Process, Quality and Productivity. Proceedings of the 5th Annual Borland International Conference. http://users.rcn.com/jcoplien/Patterns/Process/QPW/borland.html
- Harrison, Neil, and James Coplien. Patterns of Productive Software Organizations. Bell Labs Technical Journal 1(1), Summer 1996. http://users.rcn.com/jcoplien/Patterns/paper11.pdf
- Cain, Brendan, et al. Social Patterns in Productive Software Organizations. Annals of Software Engineering, December 1996. http://www.baltzer.nl/ansoft/articles/2/ase004.pdf

References and Online Resources

James O. Coplien. Organization and Architecture. In *1999 CHOOSE Forum on Object-Oriented Software Software Architecture,* pages 5-1 - 5-25, March 1999. Bern, Switzerland, Swiss Informaticians Society. A keynote on the architectural impact of organizations. http://www.bell-labs.com/user/cope/Talks/Arch/CHOOSE99/.

Organisatorisk Agility Program, http://www.nordija.dk/da/Konsulentydelser/OrganisatoriskAgility.html

Neil B. Harrison and James O. Coplien. Patterns of Productive Software Organizations. *Bell Labs Technical Journal,* 1(1):138-145, Summer (September) 1996. A good summary paper on the techniques and findings in the organizational pattern work. http://www.lucent.com/minds/techjournal/summer_96/paper11/.

James O. Coplien, Neil Harrison, and Gertrud Bjørnvig. Organizational Patterns: Building on the Agile Pattern Foundations. http://www.cutter.com/offers/orgpatterns.html. Free, but requires signup.

James O. Coplien. A Development Process Generative Pattern Language. In James. O. Coplien and Douglas C. Schmidt, editors, *Pattern Languages of Program Design,* chapter 13, 183-237. Addison-Wesley, Reading, MA, 1995. http://www.easycomp.org/cgi-bin/OrgPatterns.

References, continued.

Gabriel, R. Patterns of Software: Tales from the Software Community. New York: Oxford University Press, 1998. For the case study Cope presented. See the chapter on the re-engineering of ParcPlace Systems.

Neil B. Harrison. Organizational Patterns for Teams. In John Vlissides, James O. Coplien, and Norman L. Kerth, editors, *Pattern Languages of Program Design 2*, chapter 21, 345-352. Addison-Wesley, Reading, MA, 1996.

Brendan G. Cain and James O. Coplien. A Role-Based Empirical Process Modeling Environment. In *Prodeedings of Second International Conference on the Software Process (ICSP-2),* pages 125-133, February 1993. Los Alamitos, California, IEEE Computer Press.

Brendan G. Cain, James O. Coplien, and Neil B. Harrison. Social Patterns in Productive Software Organizations. In John T. McGregor, editor, *Annals of Software Engineering,* 259-286. Baltzer Science Publishers, Amsterdam, December 1996.