



# Beyond Agility: Organizational Patterns of Successful Software Teams

---

Neil B. Harrison  
Utah Valley State College  
[harrisne@uvsc.edu](mailto:harrisne@uvsc.edu)

With contributions from James O. Coplien

# Agile Machismo Points



- Calculate your agility points:
  - Does your entire team sit within 10 meters of each other? (10 points)
  - Have you delivered to the user in the last 6 months?
    - Twice, AND they have used it (add 100 points)
    - Not at all (subtract 200 points)

- If your score is positive:
  - Divide it by the number of people in your team (including Scrum Masters, etc.)
- If it is negative:
  - Multiply by number of people on your team
- If it is negative:
  - Multiply (again) by the number of Certified Scrum Masters you have.

- Compare with all your friends
- Highest score is the winner!
  
- Courtesy of Alistair Cockburn
  - [http://alistair.cockburn.us/index.php/Agile\\_machismo\\_points](http://alistair.cockburn.us/index.php/Agile_machismo_points)
- See also:
  - <http://cgi.nordija.com/AgileMetrics.cgi>

# Agile Machismo???



- There is much hype surrounding Agility
- It is “good” to be Agile...
  - But why?
- In this talk, we try to cut through the hype, and get back to basics

# The Agile Manifesto



We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

# Measuring Compliance...



- On a scale from 1 to 10, how closely do you follow the Agile Manifesto?
- What measures did you use?
- What does your score tell you?
  - How closely you follow the Agile Manifesto!
  - Does it say how well you develop software?

# What is the Goal?



- The Agile Manifesto doesn't say anything about:
  - Actually delivering code
  - Satisfying the customer
  - Making a profit
- In theory, you can be perfectly Agile, and perfectly unproductive...

# Is There More to it?



- Agility means much more than that!
- Of course we want to deliver working software to our customers!
- Of course we want to satisfy them!
- Note that customer satisfaction, delivering product are basic values.
- Perhaps the agile manifesto didn't say them because they seemed so universal...

# Is Agility Your Goal?



- But so-called agile practices have become a goal.
- If you do these practices, you are agile
  - Therefore you are good
- But is that really the case?
  - Or is it something deeper?
- Let's look at a few practices...

# Agile Practices...



- Frequent deliveries to customers
  - As of three weeks ago, Microsoft was making **weekly** deliveries of Vista to certain key customers.
  - (BTW, Microsoft appears to be using customer acceptance testing in the style of XP...)

# Agile Practices ...



- Frequent Integration
  - Microsoft used daily builds during development of Windows NT
    - Sources: ICSE 1995 proceedings, "Showstopper", by G. Pascal Zachary

# Agile Practices ...



- Responding to Customer Needs
  - 1625: The Swedish ship Vasa
    - Sank on her maiden voyage because she was rendered unseaworthy due to changes requested by her customer, the king.
    - Ok, they didn't do frequent deliveries.
    - But they did practice emergent design.
    - And customers did acceptance testing (may they rest in peace.)

# Agile Values ...



- Fun ...
- Ron Jeffries\* reported that there are projects that are not appropriate for XP, but he wouldn't want to work on them (OOPSLA 2000 panel)
  - But the Space Shuttle software team reports having a great deal of fun ...

\* note: it may actually have been Martin Fowler who said it.

# What's Your Goal?



- Is it to be Agile?
  
- Rock climbing:
  - What do you expect from your belayer?
    - To give encouragement?
    - To keep the tension right?
    - To pay close attention to me?
    - To catch me when I fall!
  
- Agility, is a means to an end, not the end itself.

# Beyond Agility



- Why do you practice agile software development?
  - (Probably) because you believe that it will help you produce a better product, in less time.
  - In other words, you have deeper goals and values that drive this behavior.
  - These values shape us, and help us deliver software.

# What makes Agility work?

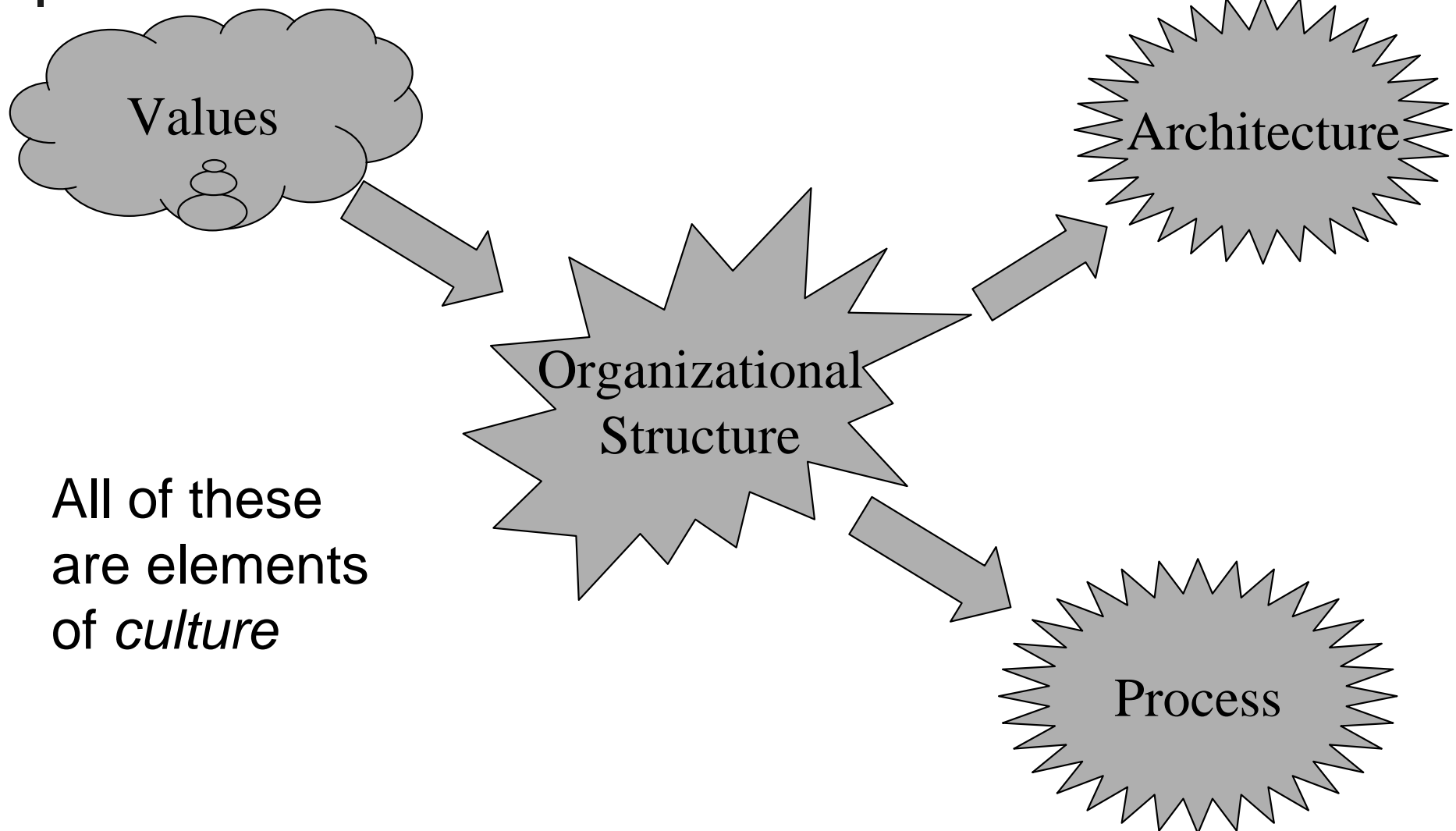


- We have studied about 100 software organizations.
- What makes some succeed and others fail?
- We have found patterns in successful software organizations
  - We see many of these practices in agile methods
  - But they transcend agility
  - There are many very successful organizations that are not "agile"
  - ... and many unsuccessful agile organizations
- These patterns are rooted in the values, structures, and practices of the organization – its culture.

# Organizational Values

- What is your organization's value system?
- What does your organization want to be?
- What are people rewarded for?
- Values drive
  - Organizational structure
  - Which drive
    - Processes

# The Chain of Influence



# Studying Organizations



- How do you study values?
  - Values are often closely held; not readily revealed
- How about Process, then?
  - WHICH Process?
  - The official process, or the one people do?
  - Processes are unstable
- You study structure instead
  - And that gives you a window into the values

# Elements of Structure



- We studied roles, responsibilities, and communication among roles
  - Role-play simulation
  - Quantitative and qualitative analysis of data

# Foundations of Effectiveness



- We identified patterns of practices and structures of highly effective teams.
- Many of these foundations underlie agile software practices
- We documented them in a book, "Organizational Patterns of Agile Software Development"
- There are about 93 patterns.
  - I'll discuss some of the most important patterns

# Foundations of Effectiveness



- Structural elements at the heart of effective teams:
  - Personal Competence
  - Community of Trust
  - High Communication, Lean Organization
  - Architectural Integrity
  - Supportive, Disciplined Processes
  - Technical and Management Support

# Personal Competence



- There is no substitute for really good people
- Considered the most influential productivity factor in COCOMO
- The key is keeping your best people
- The best organizations find ways to keep their best people happy
  - COMPENSATE SUCCESS

# COMMUNITY OF TRUST

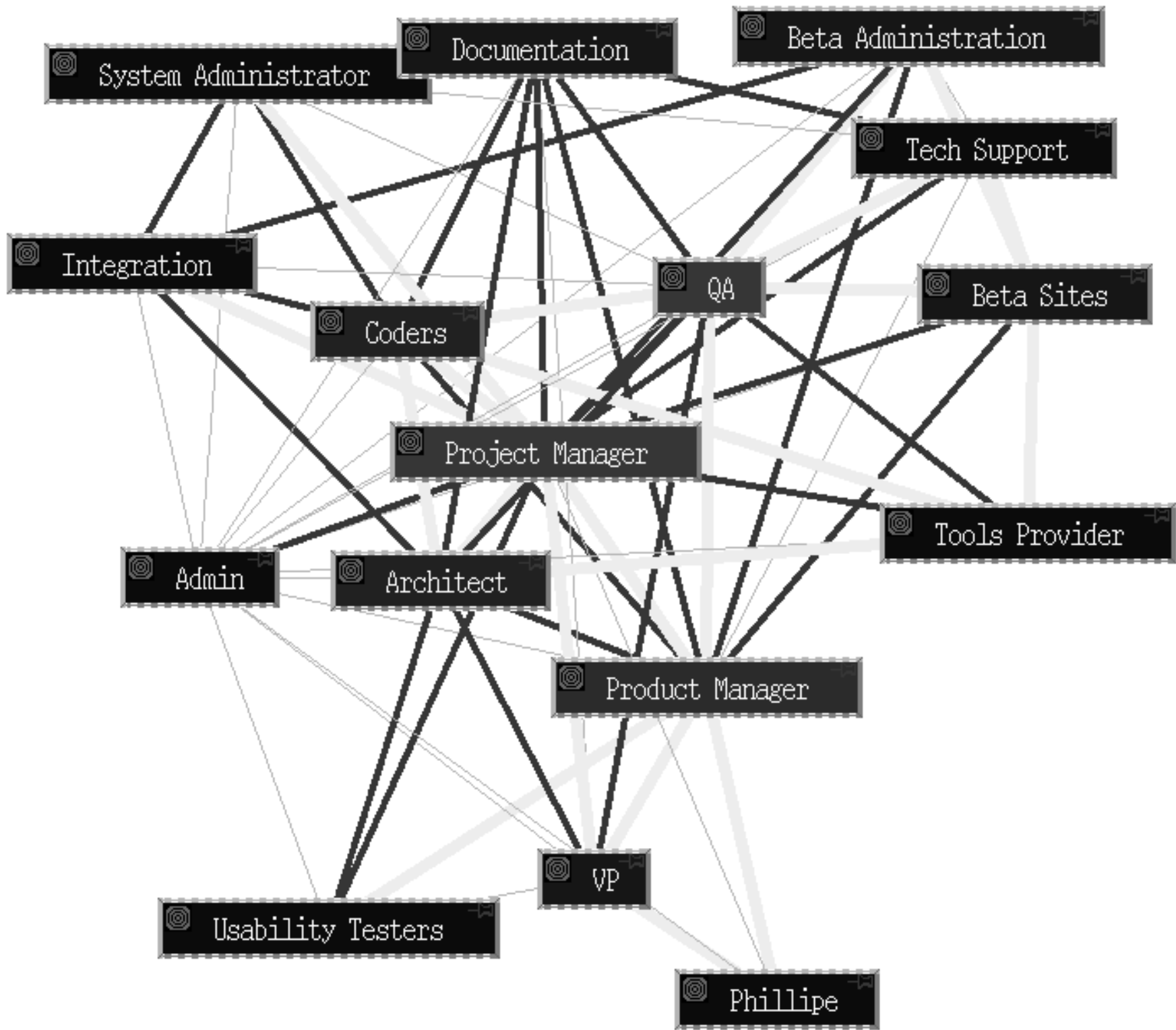


- Your team must trust each other
  - This is a basic organizational pattern
  - Without trust, you cannot hope to be effective in software development
    - ... and your best people will leave
- Trust is a characteristic of the culture that is built up over time
  - ... but can be destroyed in an instant

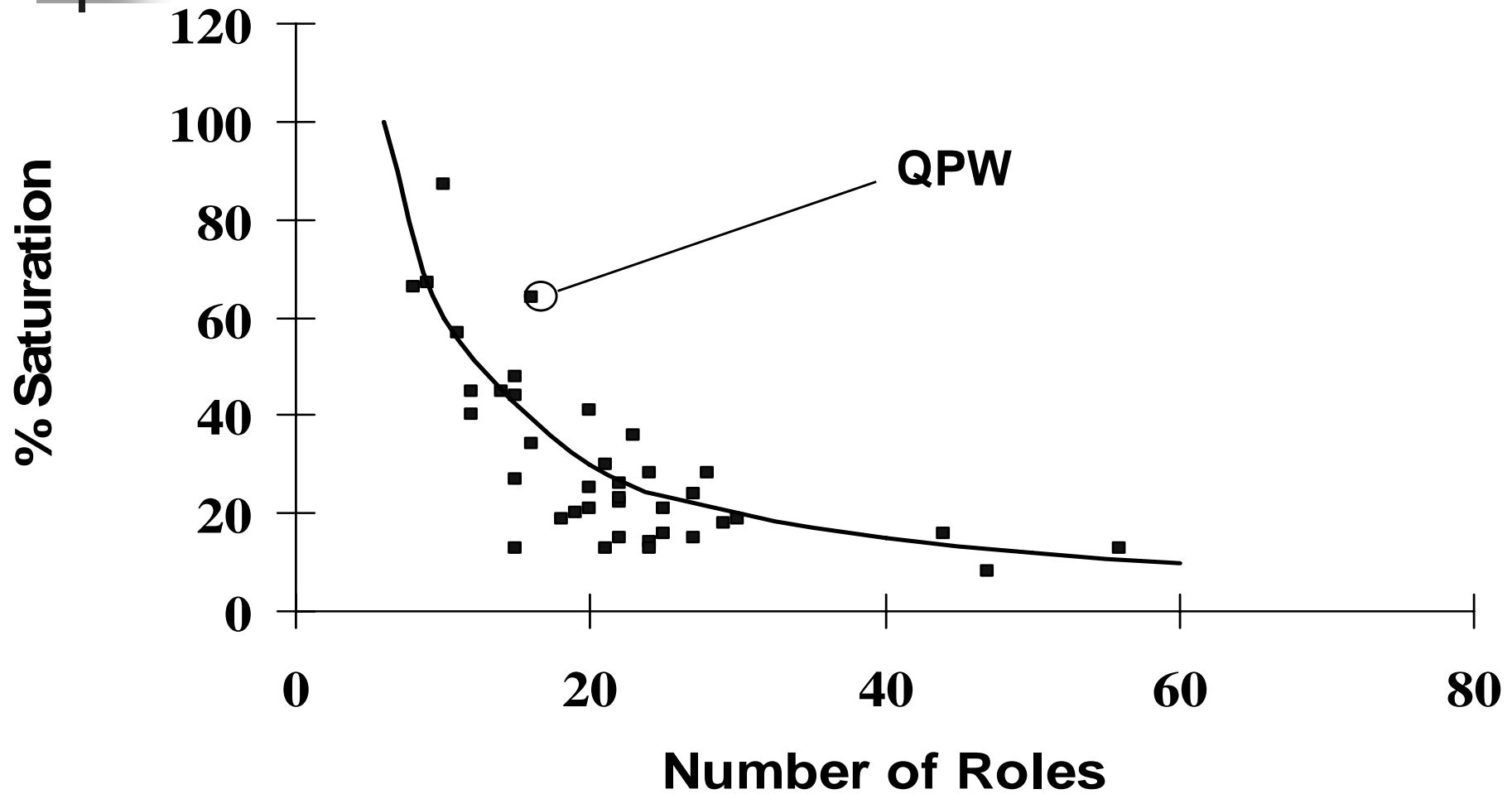
# High Communication



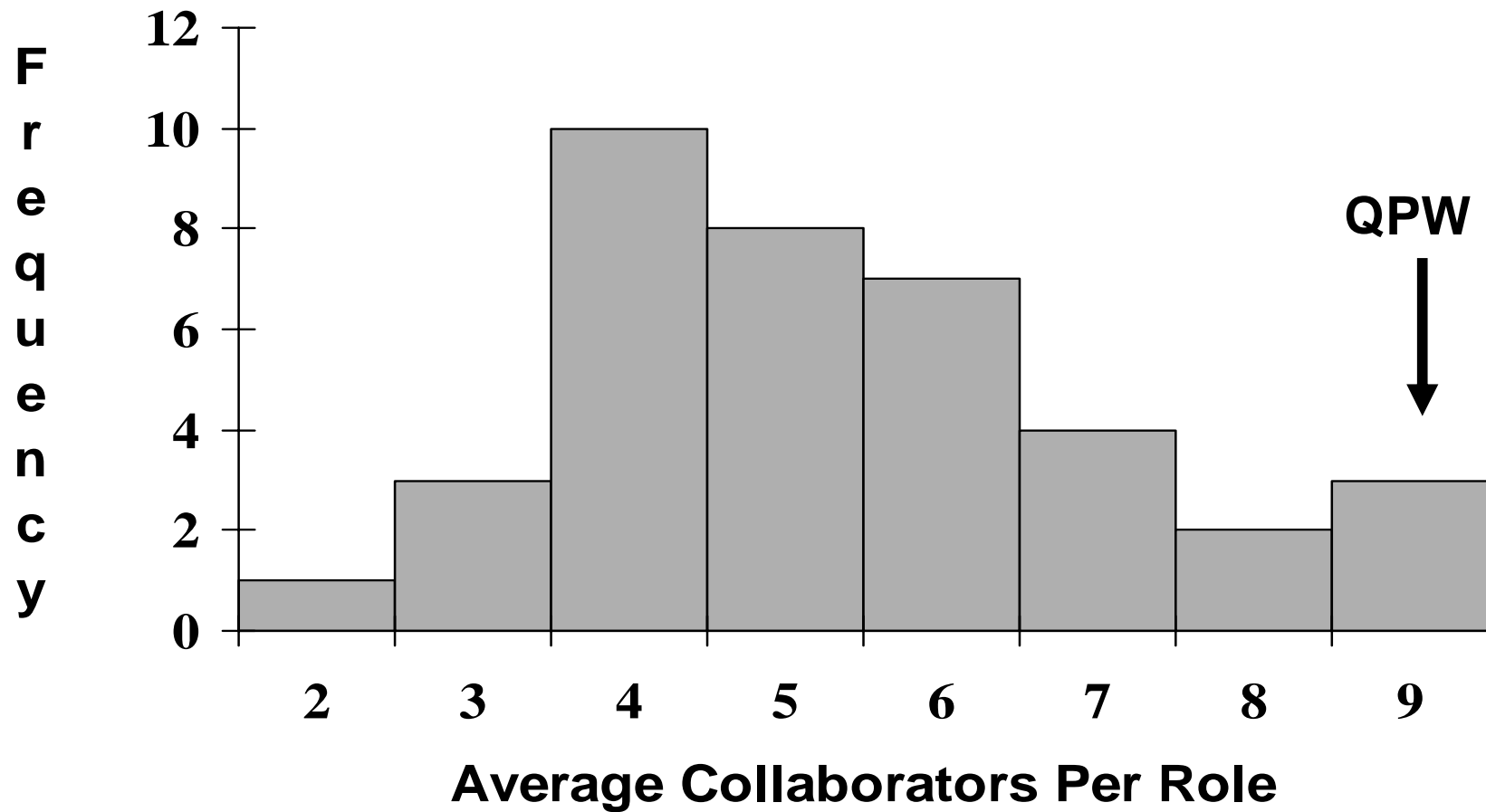
- Effective teams communicate with each other – a lot!
- Information flows quickly, through many channels
  - HALLWAY CHATTER
  - THE WATER COOLER
- Nobody is left out
  - ENGAGE QUALITY ASSURANCE
- In particular, communicate with customers
  - ENGAGE CUSTOMER



# Communication Saturation



# Communication

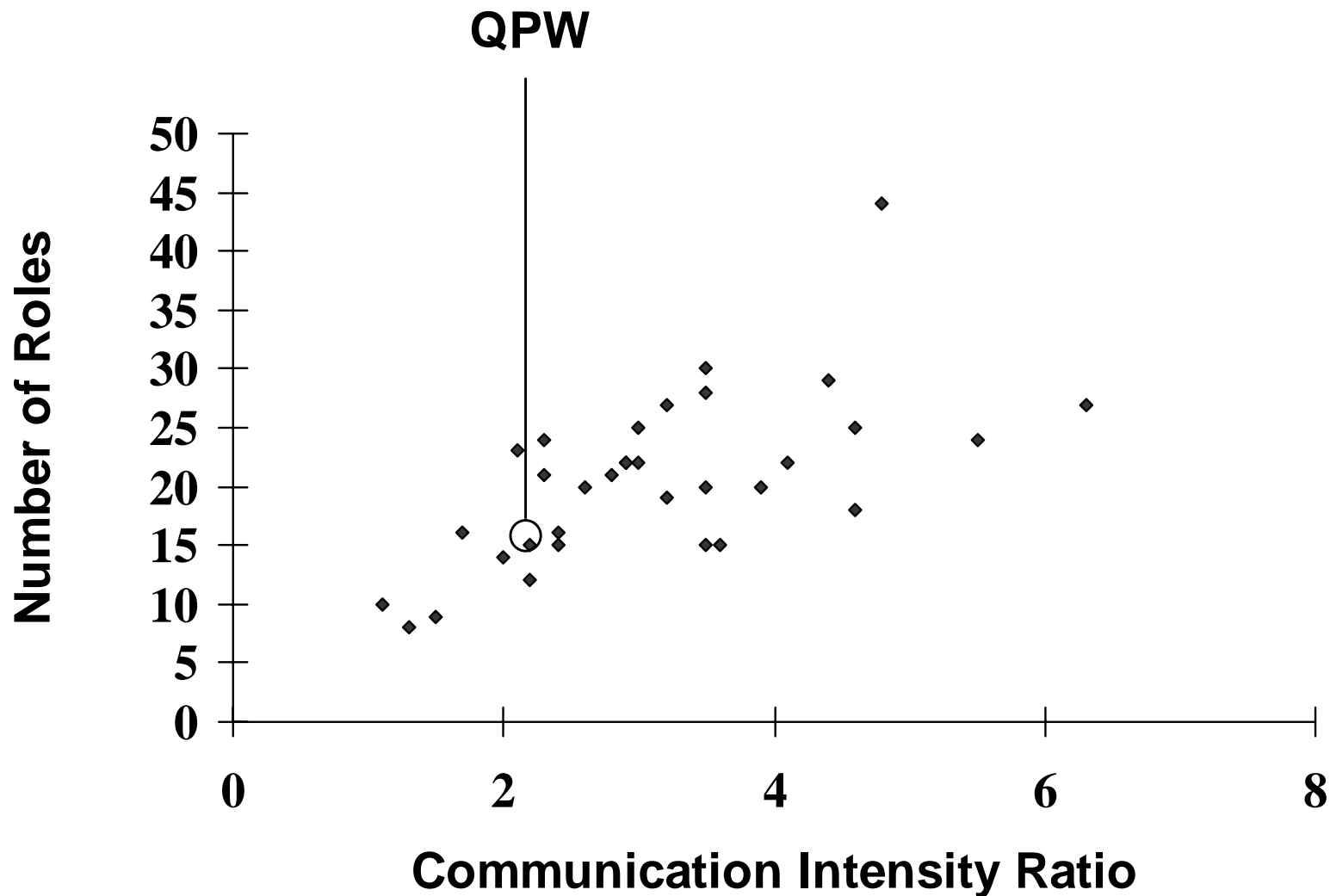


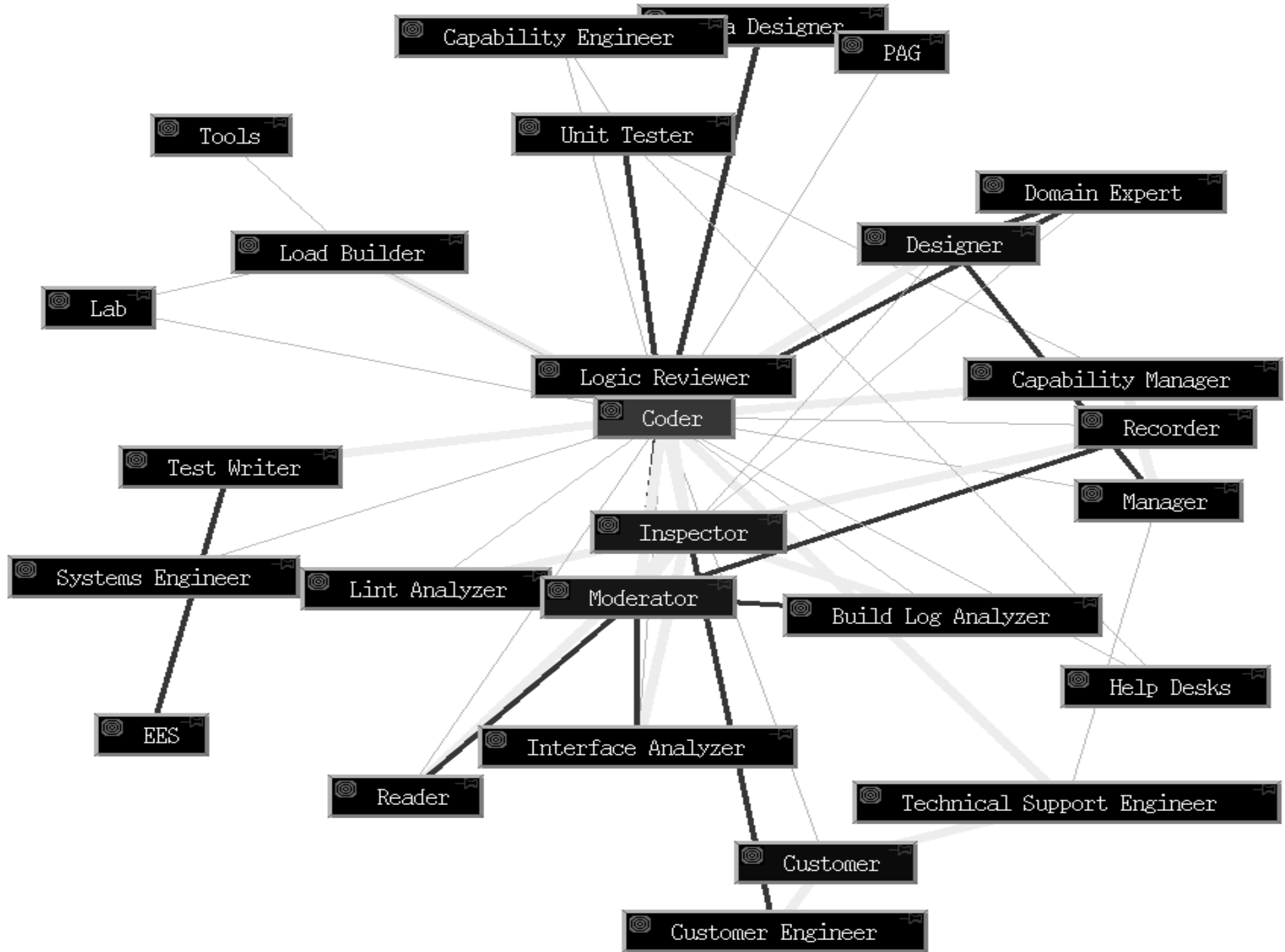
# Lean Communication



- But high communication can be expensive.
  - It can distract from the work at hand
- Balance the cost of communication by keeping a lean organization:
  - FEW ROLES
  - PRODUCER ROLES
  - DISTRIBUTE WORK EVENLY

# Distribute Work Evenly





# Architectural Integrity



- Many agile teams report having to redo their entire design two or three iterations into the project.
  - The design that emerges isn't the one they want to keep.
- If you expect your project to support evolution, spend time on the architecture.
  - It doesn't have to be a lot of time
  - And keep the integrity of the architecture
  - This is especially important for effective design of the all-important non-functional attributes
- Nearly all successful teams had a strong architecture:
  - ARCHITECT CONTROLS PRODUCT
  - ARCHITECT ALSO IMPLEMENTS

# Discipline & Support



- You need discipline!
  - Software development is a team sport...
  - Not a team of Cowboy Programmers
  - Agile Practices are highly disciplined
- Effective teams use practices that help the developer be effective – and happy:
  - DEVELOPER CONTROLS PROCESS
  - PROGRAMMING IN PAIRS
  - Managerial support: FIREWALLS

# Developer Controls Process



- Example: Stand-up Meetings
- Helps the developer be more effective
- But can be misused

# Technical Support



- It's not just how many people talk to each other. You need technological support:
  - Adequate configuration management
  - NAMED STABLE BASES
  - INCREMENTAL INTEGRATION
- Why is this an organization issue?
  - People and code are intertwined.
  - Conway's law states that the architecture and the organization tend to mirror each other.
  - There is a similar interplay between people and the infrastructure.

# Project Management



- Patterns we have seen:
  - STAND-UP MEETING
  - DEVELOPMENT EPISODE
  - TEAM PER TASK
  - WORK FLOWS INWARD
- Note: We do these for the purpose of helping the work move forward smoothly.
  - In other words, don't do these because they are agile, but agile does these because they work.

# Where does Agility Fit?



- You will certainly have recognized some of these patterns.
- Many agile processes include some of these patterns.
- Most are compatible with all of them.
- While the patterns predate agile methods, that's not the point:
  - Instead, understand the basics of effectiveness.

# Following the Principles



- How many of these patterns do you use?
- Again, that is the wrong question.
- The goal isn't to use the most patterns
  - Or to have the highest agility machismo score
- Ask yourself how you are doing with regards to what really matters...

# The Bottom Line

- Organizations that have this foundation tend to be successful
  - Regardless of whether they are agile or not.
- If they don't, they have problems
  - Success comes only in spite of themselves
  - Agility won't help

# Team Introspection



- The (relatively) easy question:
  - How effective are you?
  - Measure by productivity, quality, customer satisfaction, etc.
- The harder question:
  - Why?
  - If you aren't where you want to be, what is keeping you from it?

# Incremental Improvement



- Changing an organization is difficult
  - Because you are changing the culture
- Apply changes piecemeal
  - For example, one pattern at a time
- Large-scale changes may be precipitated by a crisis
  - Crises force introspection
  - Crises cause us to question our values
- Our organizational studies have resulted in a few notable successes

# When all is Said and Done...



- What really matters is delivering the goods:
  - On time
  - High quality
- And be able to do it over and over again!
  - Without killing ourselves...

# Good Advice?

- Use the patterns, agile practices, CMMI, PSP etc., where they work for you.
- The patterns expose basic principles of organizational dynamics.
  - But still, use them according to your own situation.
- Keep your eyes on your ultimate goals.

# To Contact us:



- Neil B. Harrison
  - [harrisne@uvsc.edu](mailto:harrisne@uvsc.edu)
- James O. Coplien
  - [JOCoplien@cs.com](mailto:JOCoplien@cs.com)

# References



James. O. Coplien and Neil B. Harrison, *Organizational Patterns of Agile Software Development*. Prentice-Hall, Upper Saddle River, NJ, 2005.

<http://www.easycomp.org/cgi-bin/OrgPatterns>

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/>.

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. Available from the authors.

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.

# More References



- Coplien, James O. Borland Software Craftsmanship: A New Look at Process, Quality and Productivity. *Proceedings of the Fifth Borland International Conference*, Orlando, FL, June 1994,  
<http://www.bell-labs.com/user/cope/Patterns/Process/QPW/borland.html>.
- Gabriel, R. Patterns of Software: Tales from the Software Community. New York: Oxford University Press, 1998. 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 *Proceedings of Second International Conference on the Software Process (ICSP-2)*, pages 125-133, February 1993. Los Alamitos, California, IEEE Computer Press.
- Thomas J. Allen. Managing the Flow of Technology, MIT Press, Cambridge, MA, 1977.