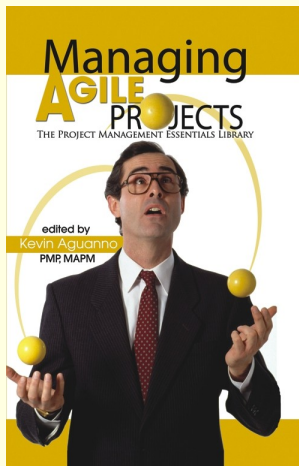


# Extreme Scrum:

200 team members in 17 cities  
on a high-risk project



Kevin Aguanno, PMP, MAPM

Editor of

*Managing Agile Projects*

# What was the System to be Built?

---

## Autoprovisioning for action and automation

- Automates the configuration and activation of:
  - Servers and other processors
  - Storage
  - Network connectivity, firewalls, and load balancers
- Software and content management to:
  - Install operating systems, middleware, customer and 3rd party applications
  - Test, add, or remove incremental changes such as fix and security patches
  - Install databases, files, and content
  - Activate change under admin or customer control
- Approach
  - Drag and drop GUI interface
  - Component to component communication via MQ and XML
  - Reliance on datastore

# The Problem

---

- Business managers had oversold a proof of concept to the senior execs
- Sold to customers under contract
- Financial penalties if not delivered by Dec. 31
- More contracts lined up in the New Year
- As of Sep. 1:
  - lots of HL design done, but design still in flux
  - no code written
  - requirements still changing daily

# Why use Agile?

---

- **Short development schedule** required immediate visibility into problems
- **Frequent requirements changes** indicated an agile approach to break out of “design churn”
- **Quality a concern**, so needed testing throughout the project, not just a couple of weeks at the end
- **Many different development organizations**, each using different development methods & terminology – need one single management method that is largely development method independent.
- **Could break the system down** into many components, each of which could be developed by a (mostly) independent team.
- **Lots of executive involvement** = active stakeholder participation

# Why Scrum?

---

With Scrum, we chose a development-method neutral management system wherein we:

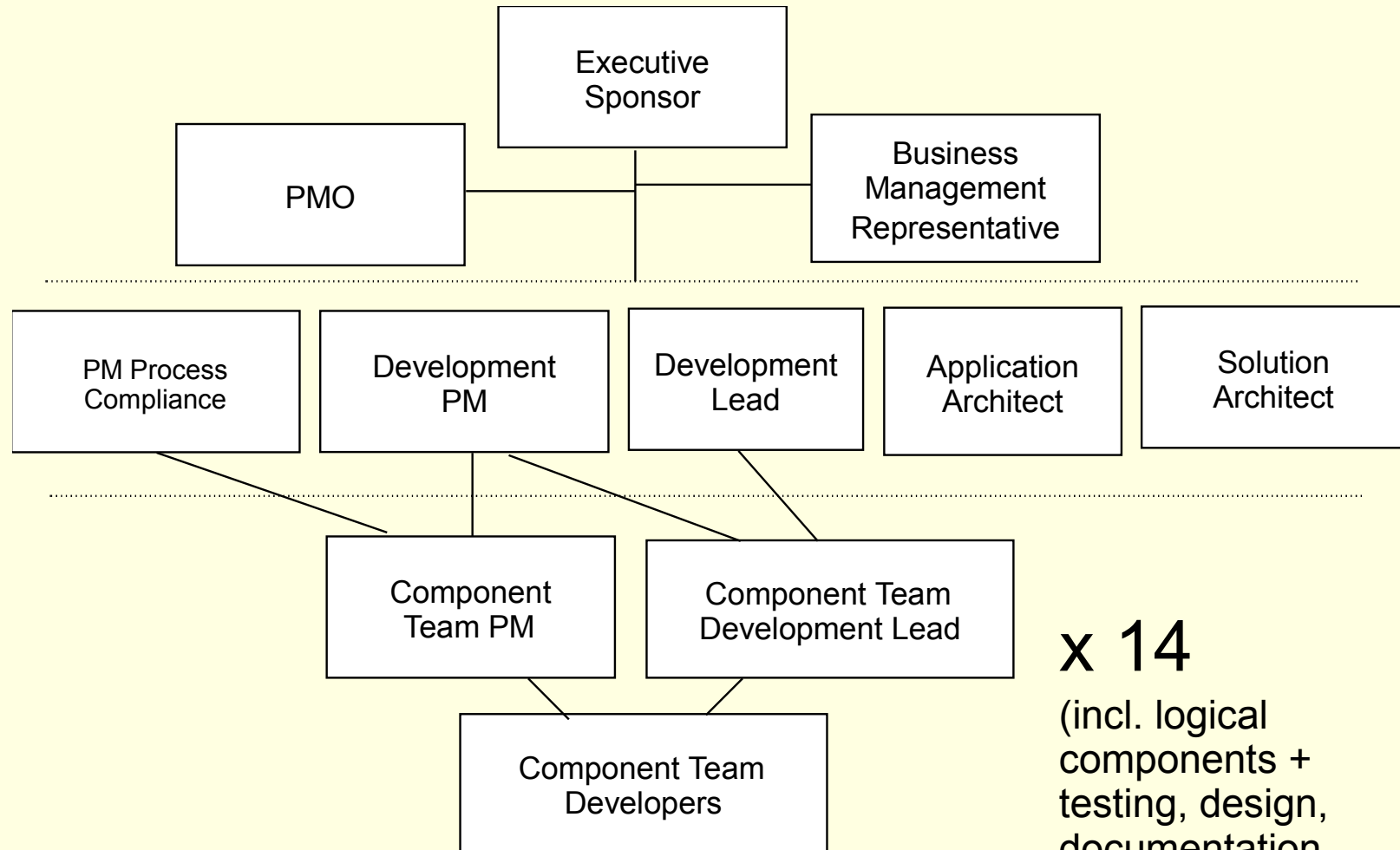
- Used iterative development to reduce risk
- Used concurrent design which increased risk, but shortened the timeline
- Divided the development team along the lines of the discrete logical components to reduce complexity
- Aligned development iterations with groups of business transactions to reduce complexity
- Conducted concurrent testing at the end of each development iteration which lowered risk and shortened the timeline

# Challenges in Scaling Scrum

---

- Geographic team distribution (North America coast to coast + UK + India) = 17 cities, 4 countries
  - Time zones
  - Lack of face-to-face meetings (see upcoming slide)
    - Miscommunications
    - Failure to gain commitment
    - Need for more robust/frequent documentation
    - Delays in communications

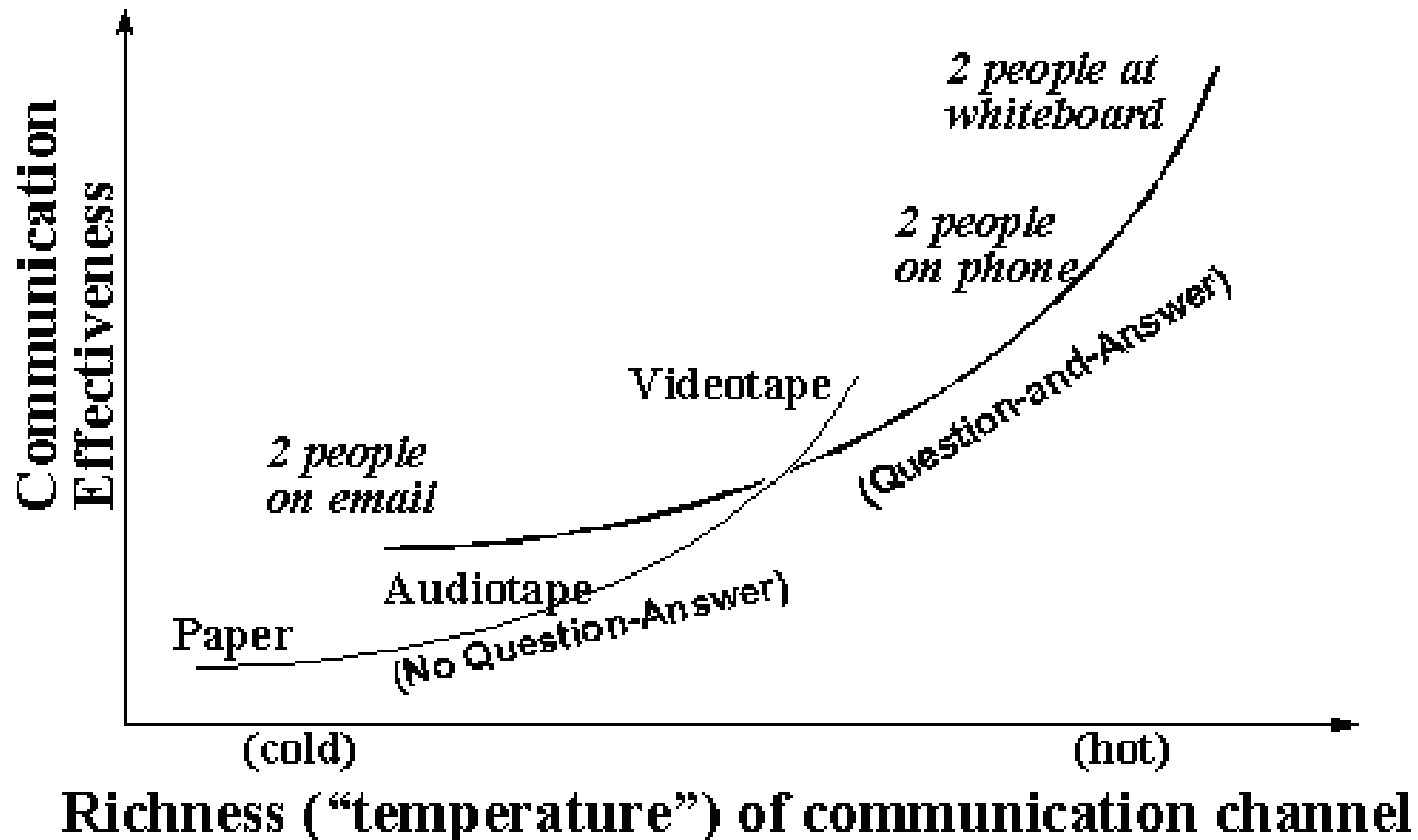
# Project Team Organization



**x 14**

(incl. logical components + testing, design, documentation, build, etc.)

# Alistair Cockburn Suggests Using High-Bandwidth Communications



# Daily Information Flow was a 4-Step Process:

---

1. Daily team Scrum meetings for each component attended by PM and Tech Lead for each component
2. PMs then had an “all PMs” Scrum meeting to resolve inter-component project dependencies and overall integration schedule, chaired by overall Development PM
3. Tech Leads had an “all Tech Leads” Scrum meeting to resolve inter-component technical dependencies (messaging standards, etc.) chaired by overall Development Lead
4. Core Team had a Scrum meeting to discuss/resolve any unresolved escalations/issues resulting from the PM and Tech Leads calls.

# Too Many Cooks in the Kitchen

---

- With a large team/complex design, had to centralize overall architecture/messaging design into a central overhead team who would be the authoritative source for message standards.
- Evolutionary design and changing requirements led to frequent database design changes. (Abstracting the database layer saved the project.)

# Not Everyone Comfortable Working in an Agile Environment

---

- Some people ***need*** the certainty and structure that comes from deterministic methods – these people will not be comfortable working in an agile project
- We had to replace some team members (and one entire component team) who could not adapt to the new paradigm
- Timeboxing requiring scope adjustments to meet the firm deadline was the hardest element for some people to adjust to.

# Documentation was Less than Agile

---

- Due to the large, distributed team, face-to-face communication was mostly replaced with emails and documents.
- Even though we kept them lightweight, preparing and updating the internal working documents slowed down the development team; however, there were no viable options with this large dispersed team.
- Lastly, this project was audited as part of a CMMI Level 3 assessment during the project, requiring additional team members put in place by the business sponsor solely to create documents required to pass the audit. No one used the docs, but we passed the audit.

# Net Results?

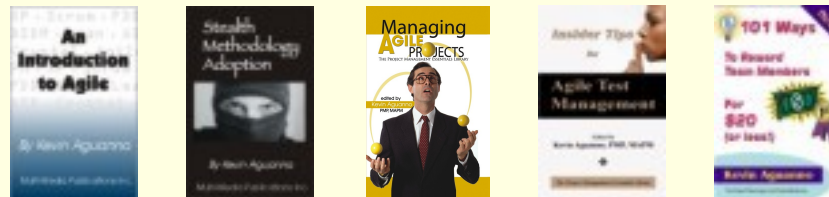
---

- In 5 months:
  - Was able to deliver contractual minimums plus incorporate customer high-priority changes.
  - Over 1 million lines of code (more than 600k SLOC hand-written)
  - 23 patent filings! (We did what was thought next to impossible)
  - Avoided contractual penalties.
  - Came in under the budget forecasts prepared for a team following the traditional method

# Additional Resources on Using Agile Techniques

- Many resources available via the Agile Alliance Web Site: [www.agilealliance.com](http://www.agilealliance.com)
- Kevin Aguanno (your speaker) is available for consultation at [aguanno@ca.ibm.com](mailto:aguanno@ca.ibm.com). He is the author of several books related to this subject matter (available at your local bookstore or from [www.Amazon.com](http://www.Amazon.com)):

## Books:



## Audiobooks:

