QCon

Kanban

Creating a Kaizen Culture and evolving Lean Software Engineering Solutions

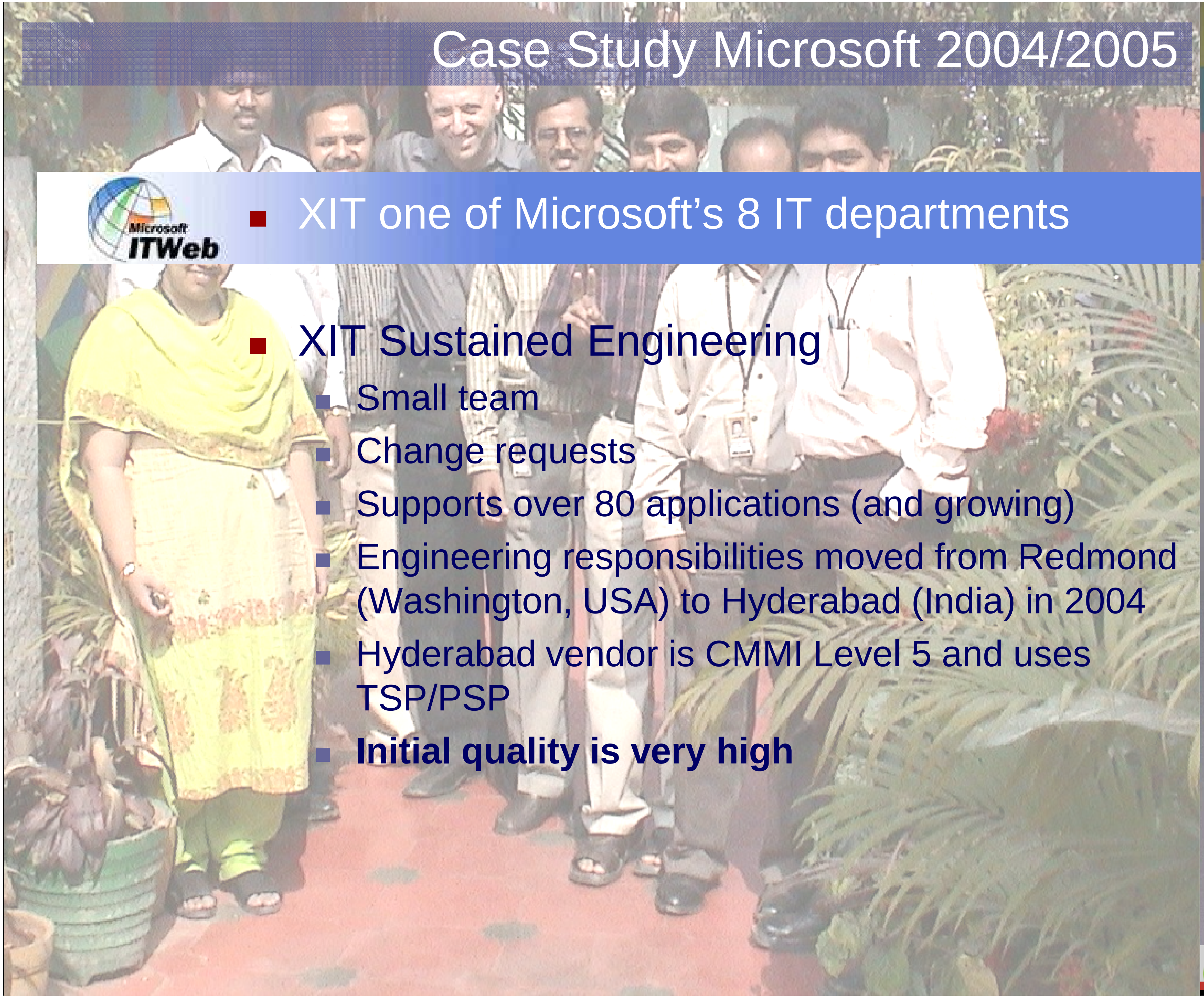
David J. Anderson
President, Modus Cooperandi,
Performance Through Collaboration

What is a kanban system?

Kanban allows us to implement my recipe for success

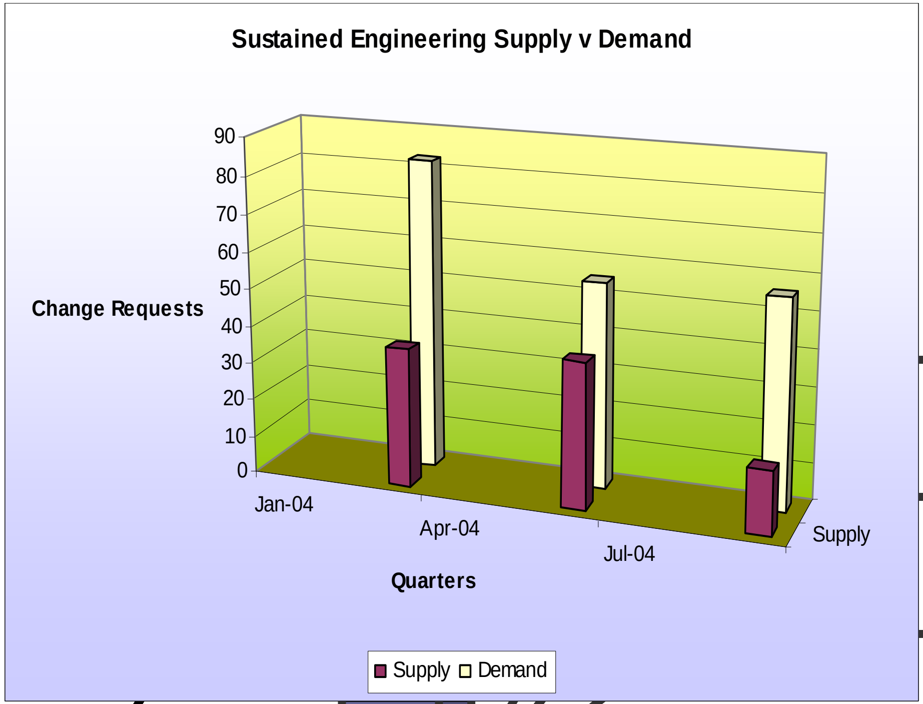
- Focus on Quality
- Reduce (or limit) Work-in-Progress
- Balance Demand against Throughput
- Prioritize

Case Study Microsoft 2004/2005



- XIT one of Microsoft’s 8 IT departments
- XIT Sustained Engineering
  - Small team
  - Change requests
  - Supports over 80 applications (and growing)
  - Engineering responsibilities moved from Redmond (Washington, USA) to Hyderabad (India) in 2004
  - Hyderabad vendor is CMMI Level 5 and uses TSP/PSP
  - **Initial quality is very high**

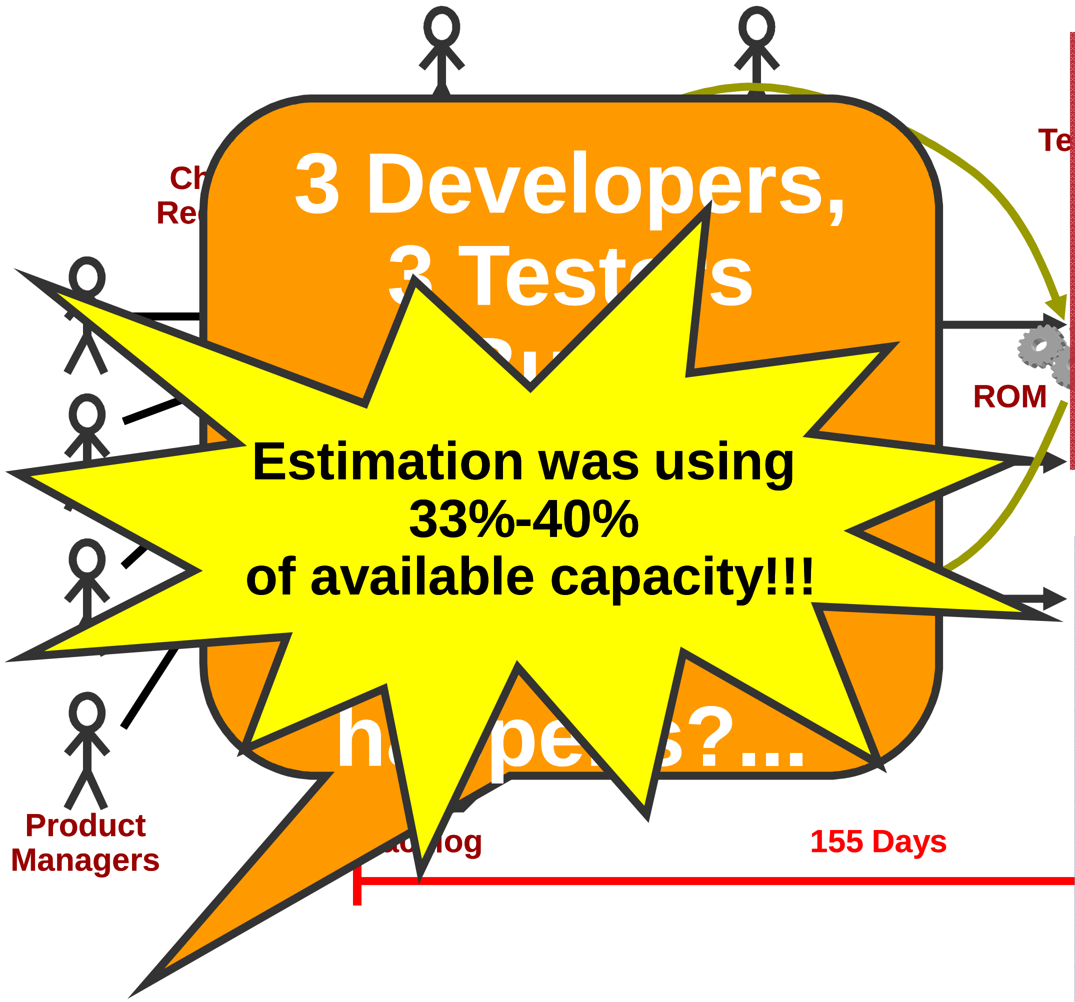
Dark Days in July 2004



Demand is outstripping supply and the queue is growing

- Manager resigns end June 2004 – open position Q3 2004
- Backlog is 80+ and growing about 20 per quarter
- Lead time is 155 days
- Customer satisfaction – lowest in IT department

Estimation (ROM) was Top Priority

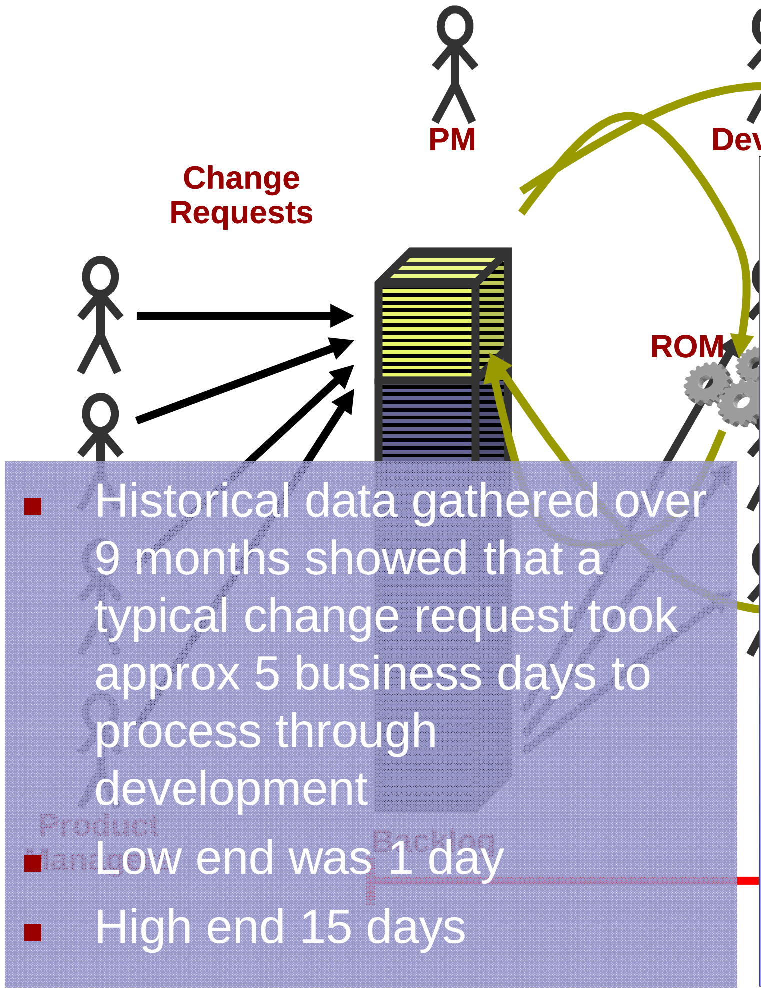


Estimation was using 33%-40% of available capacity!!!

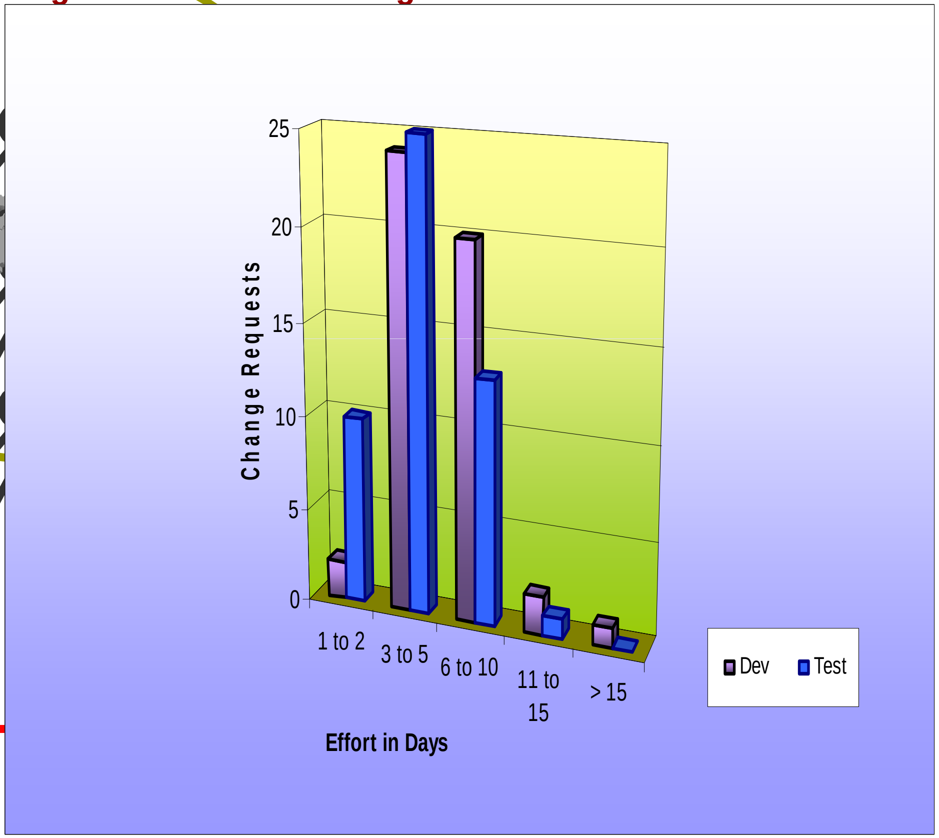
- Open and Read Source Code
- Read Application Guide
- Whole process about 1 day per developer and tester

- SLA – 48 hours to return a rough order of magnitude estimate (ROM)
- All change requests are ROM estimated
- ROMs are expedited as top priority due to SLA

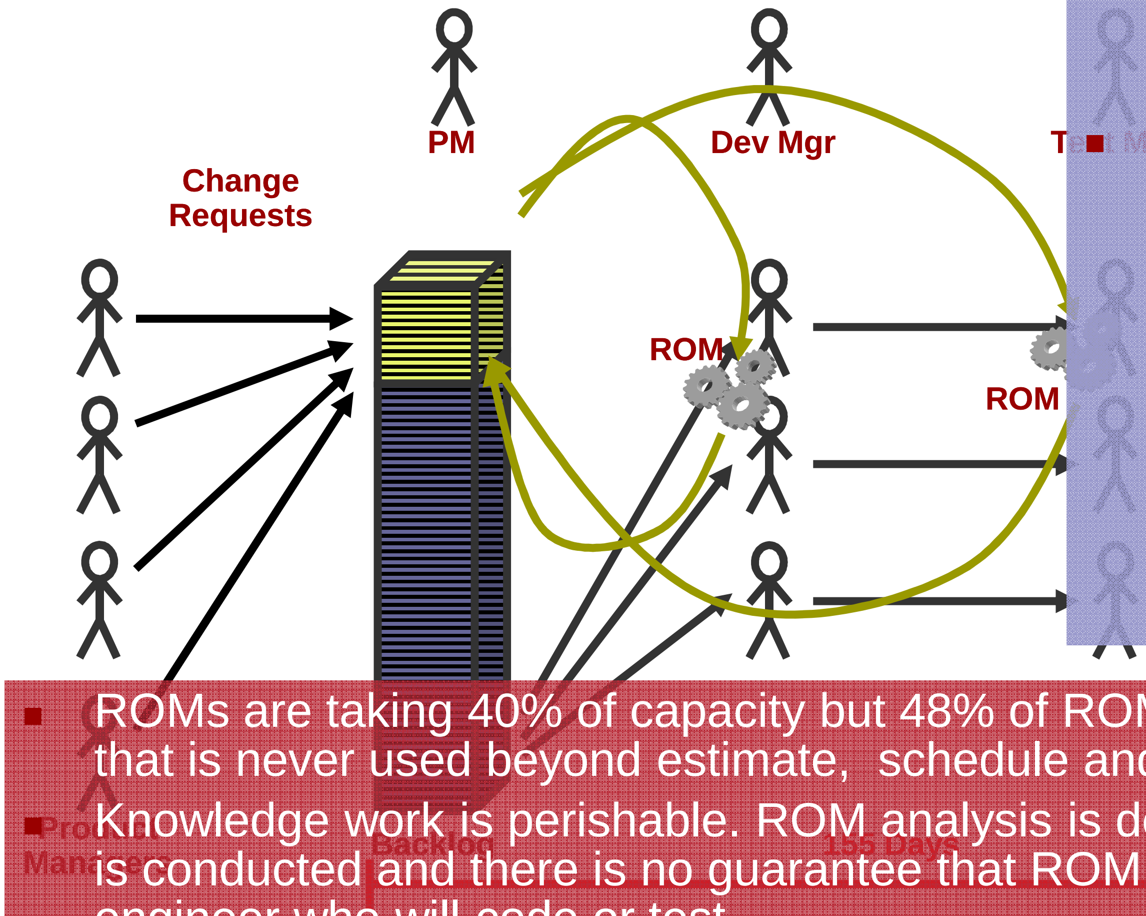
Actual effort was miniscule compared to lead time of 155 days



- Historical data gathered over 9 months showed that a typical change request took approx 5 business days to process through development
- Low end was 1 day
- High end 15 days

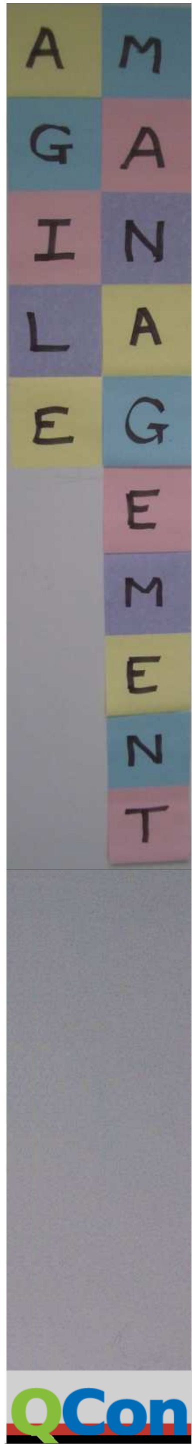


Are Estimates *muda?*

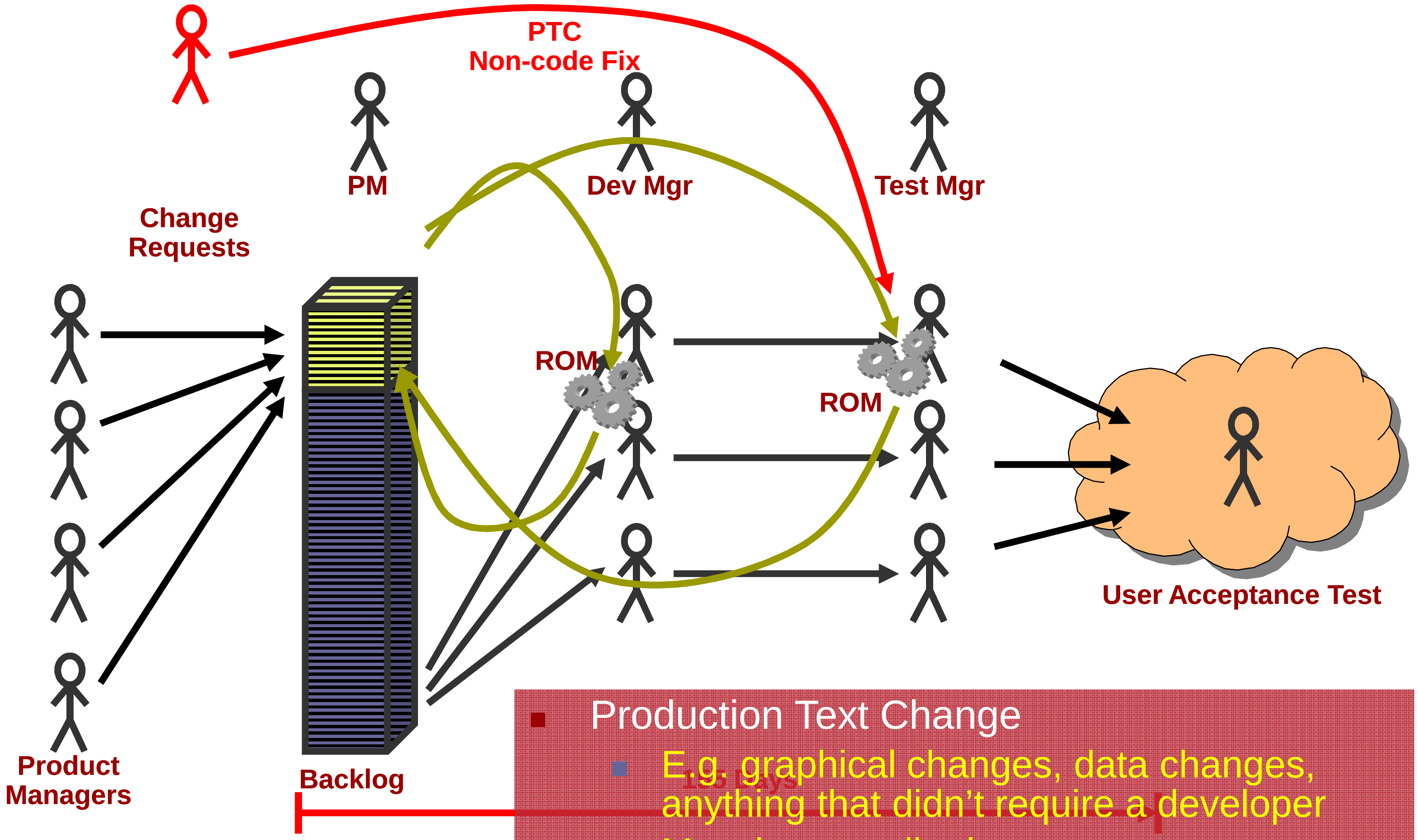


- Only 52% of requests were actually ever completed
- Other 48%
  - Too big (bigger than 15 days)
  - Too expensive (low value versus cost)
  - Overtaken by events, application decommissioned before request is processed

- ROMs are taking 40% of capacity but 48% of ROMs represent analysis that is never used beyond estimate, schedule and go/no go decision!
- Knowledge work is perishable. ROM analysis is done months before work is conducted and there is no guarantee that ROM is conducted by same engineer who will code or test.
- Conclusion – all ROMs are *muda*



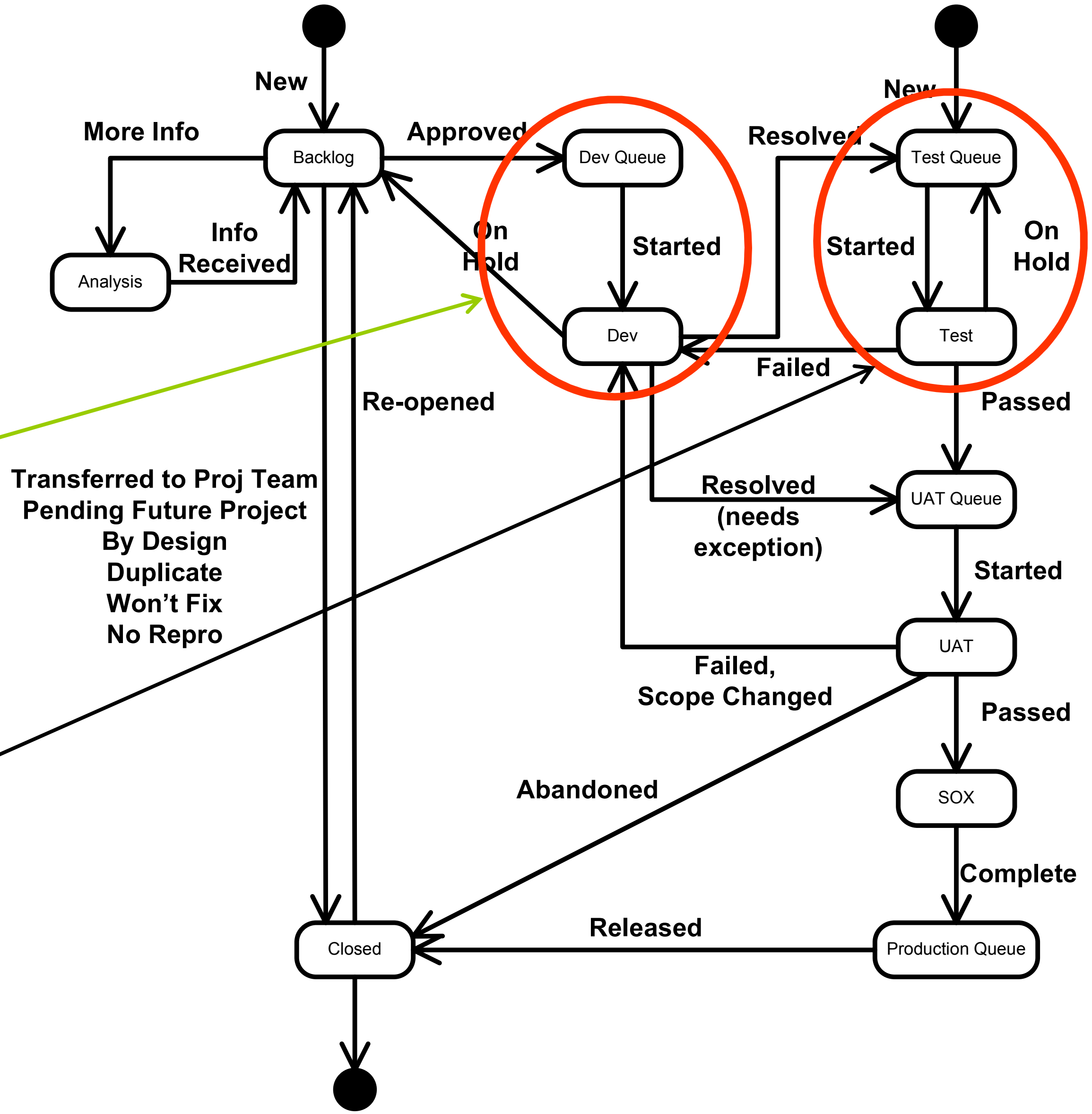
Could it get worse? Expediting



- Production Text Change
  - E.g. graphical changes, data changes, anything that didn’t require a developer
  - Must be expedited
- Need to make formal QA pass

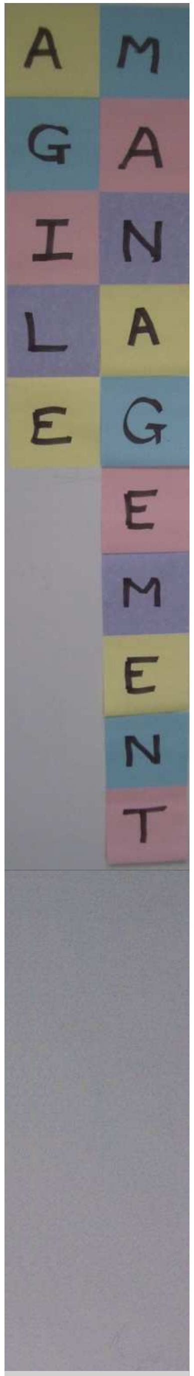
State Model

XIT Change Request



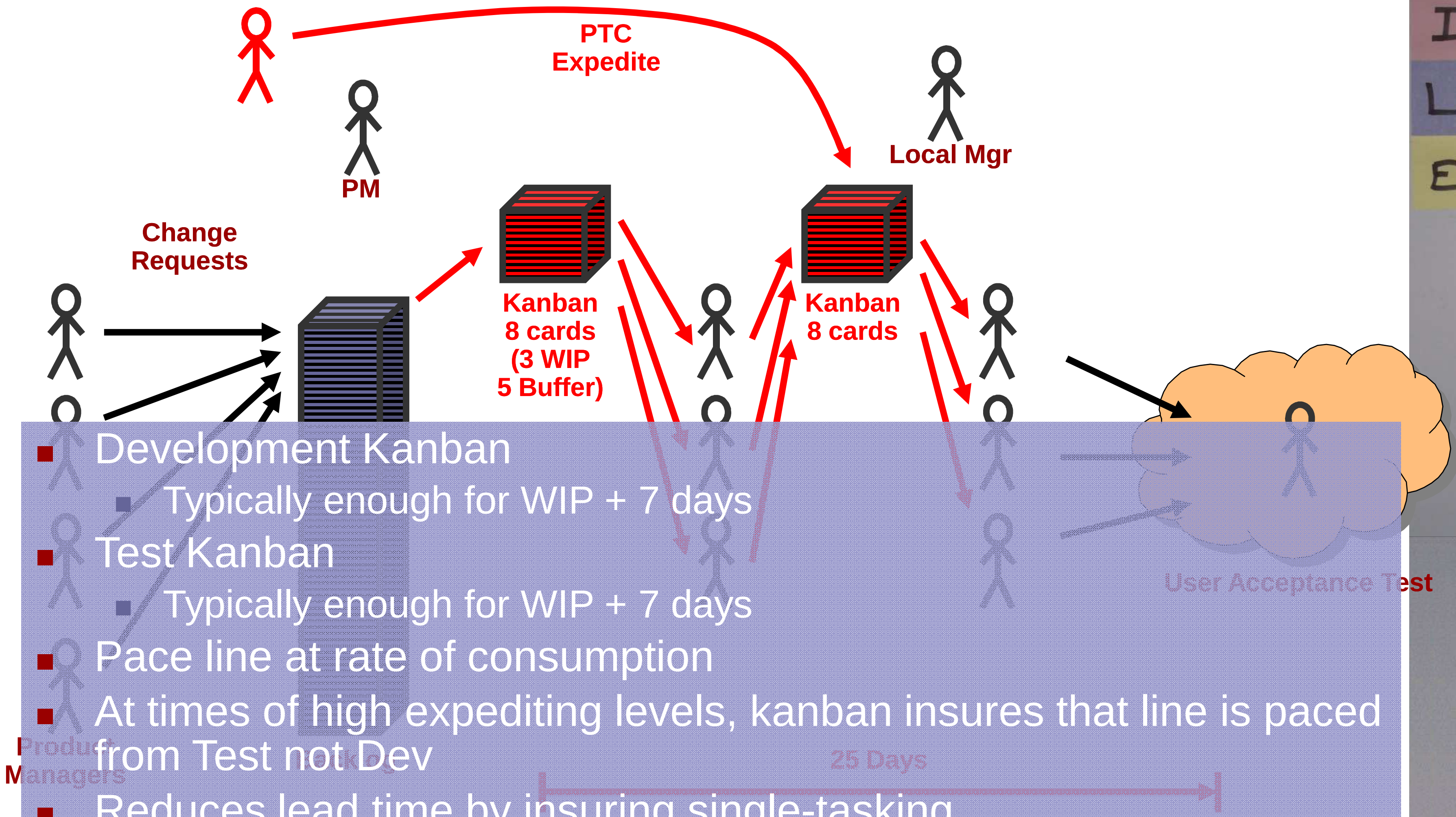
Virtual Kanban limit initially 8 = WIP + 7 days buffer

Virtual Kanban limit initially 8 = WIP + 7 days buffer



QCon

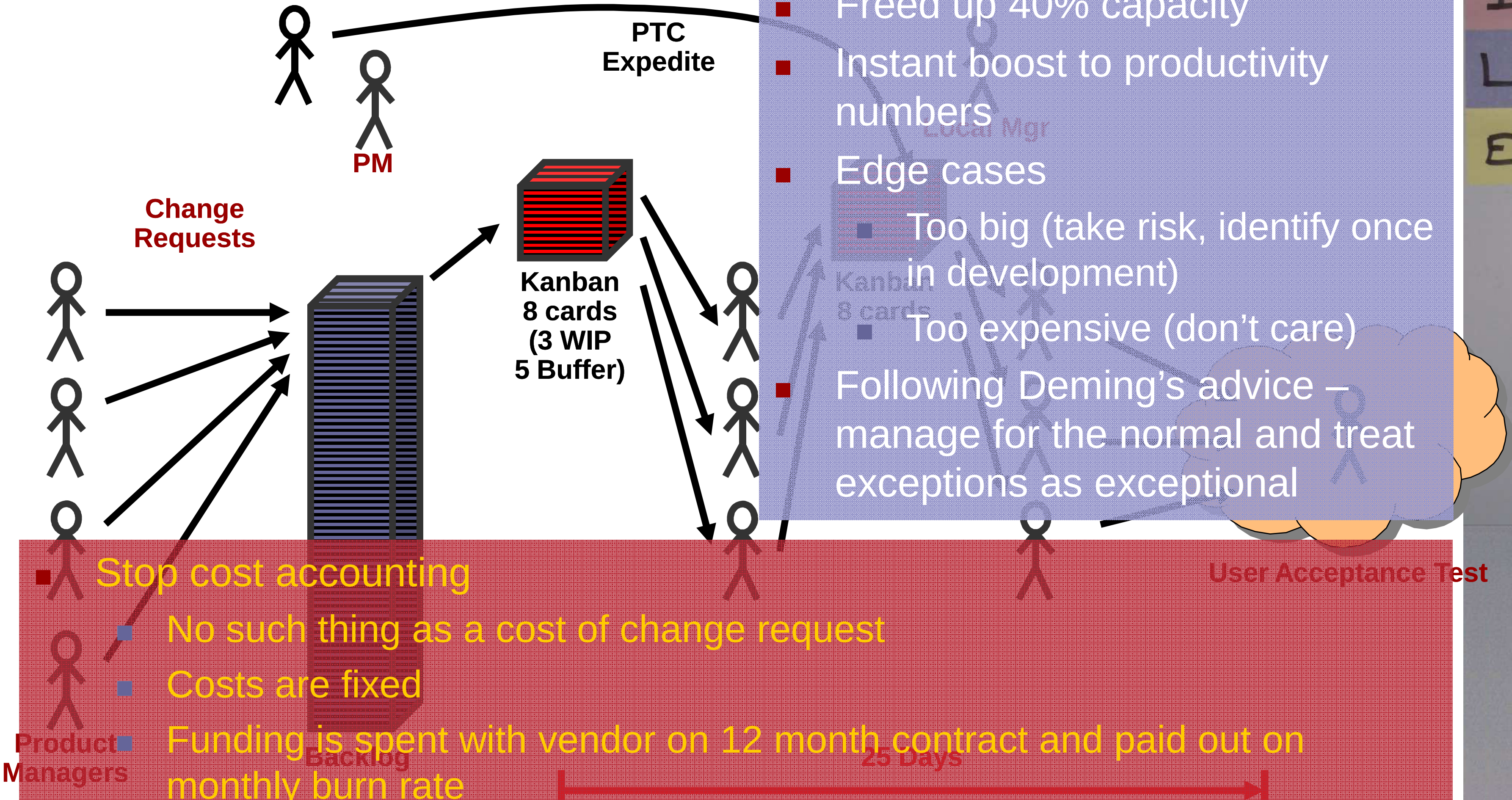
Intervention 1
Pace the Line from Development



- Development Kanban
  - Typically enough for WIP + 7 days
- Test Kanban
  - Typically enough for WIP + 7 days
- Pace line at rate of consumption
- At times of high expediting levels, kanban insures that line is paced from Test not Dev
- Reduces lead time by insuring single-tasking
- Focuses customer acutely on selection of highest priority (urgency) requests for insertion into empty buffer slots

Intervention 2 – Stop Estimating

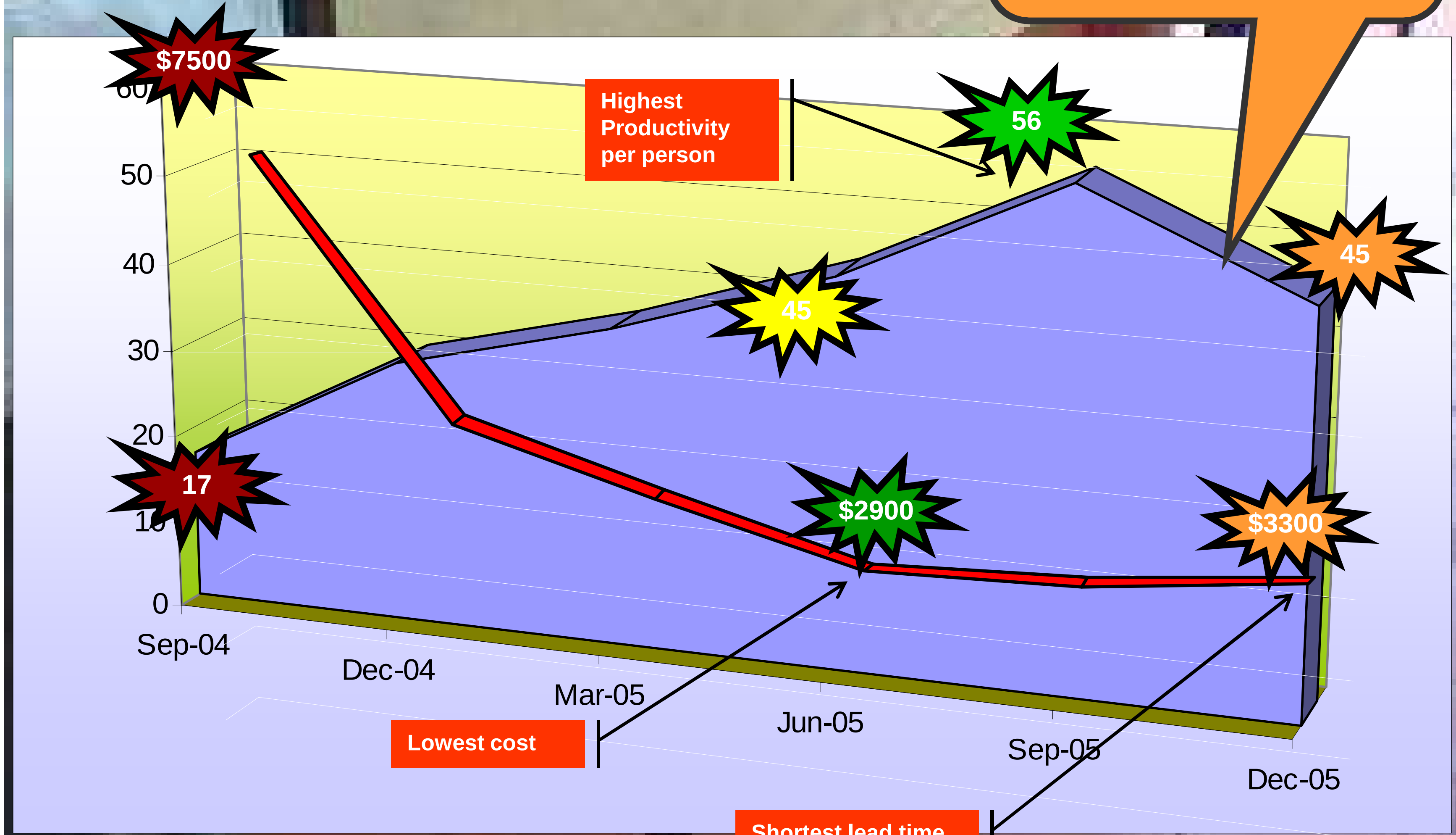
- ROM activity abandoned
- Freed up 40% capacity
- Instant boost to productivity numbers
- Edge cases
  - Too big (take risk, identify once in development)
  - Too expensive (don't care)
- Following Deming's advice – manage for the normal and treat exceptions as exceptional



- Stop cost accounting
  - No such thing as a cost of change request
  - Costs are fixed
  - Funding is spent with vendor on 12 month contract and paid out on monthly burn rate
- All changes would be treated equally for cost purposes
  - Based on average of 5 business days through development

Throughput and Cost

Zero Backlog achieved
Some idle time
Lowers metrics

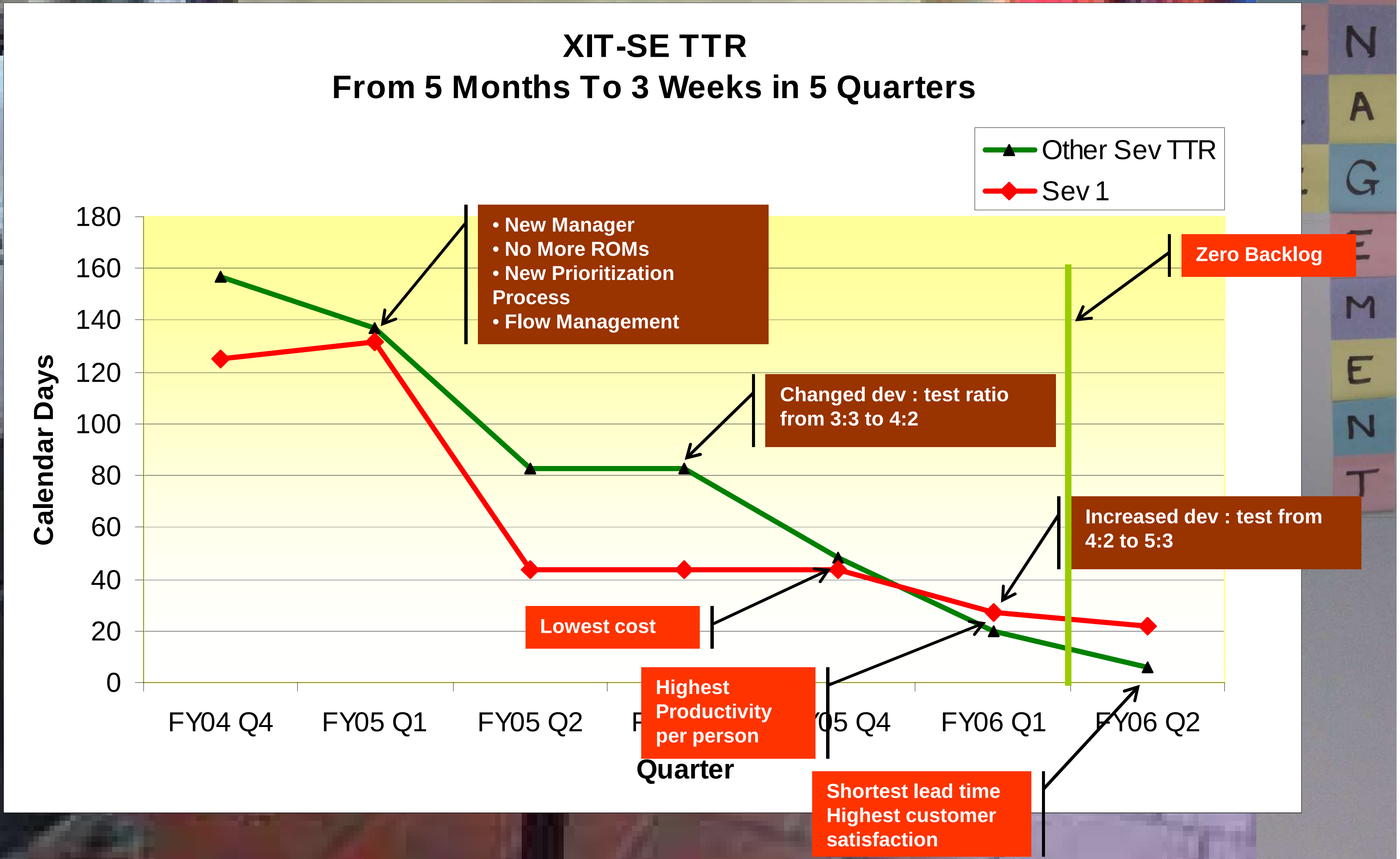


Highest Productivity per person

Lowest cost

Shortest lead time
Highest customer satisfaction

Why Lead Time is the best metric !



- New Manager
- No More ROMs
- New Prioritization Process
- Flow Management

Changed dev : test ratio from 3:3 to 4:2

Zero Backlog

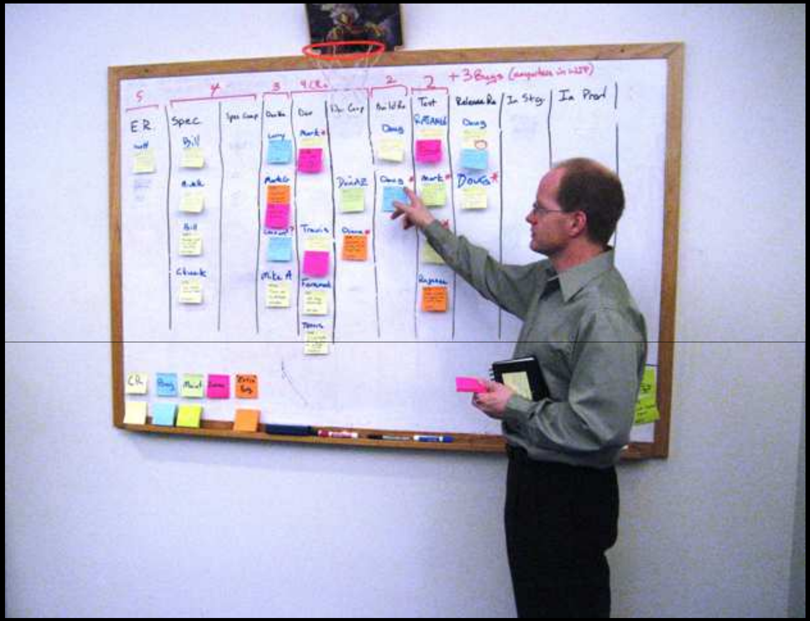
Increased dev : test from 4:2 to 5:3

Lowest cost

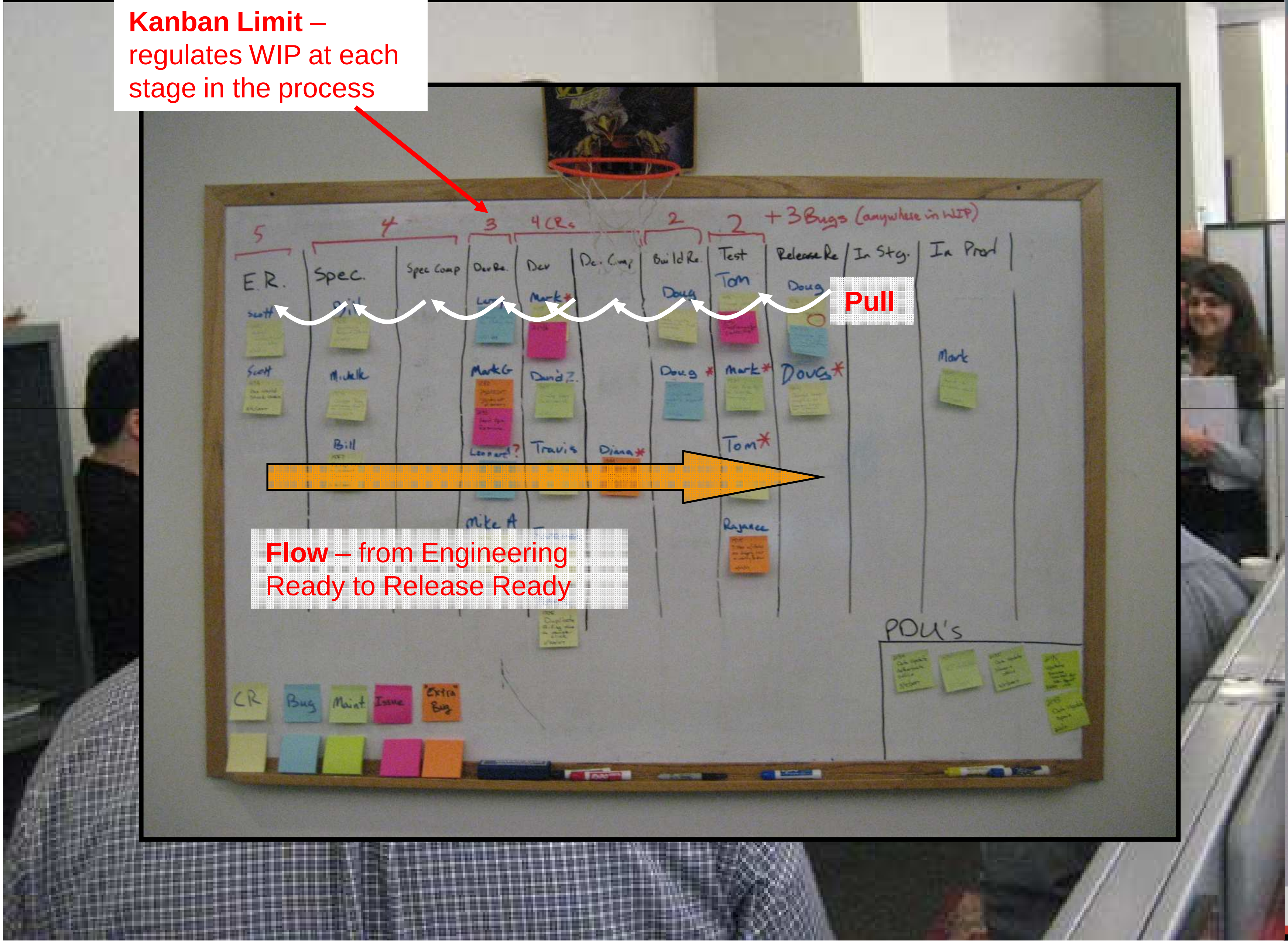
Highest Productivity per person

Shortest lead time
Highest customer satisfaction

At Corbis in December 2006, we implemented a detailed kanban system for sustaining engineering



Kanban limits create a pull system and white board provides visualization of flow through to delivery

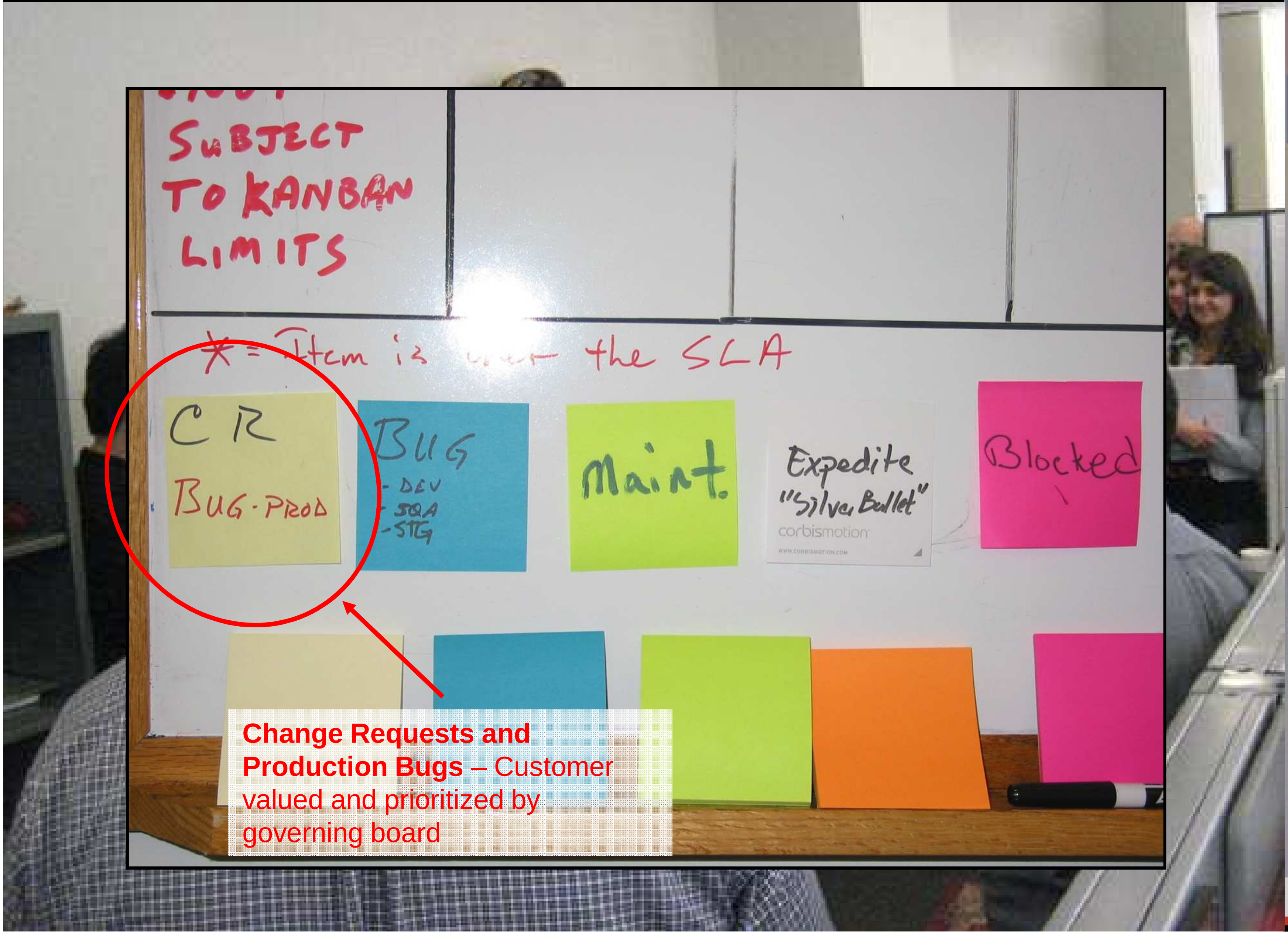


Kanban Limit – regulates WIP at each stage in the process

Pull

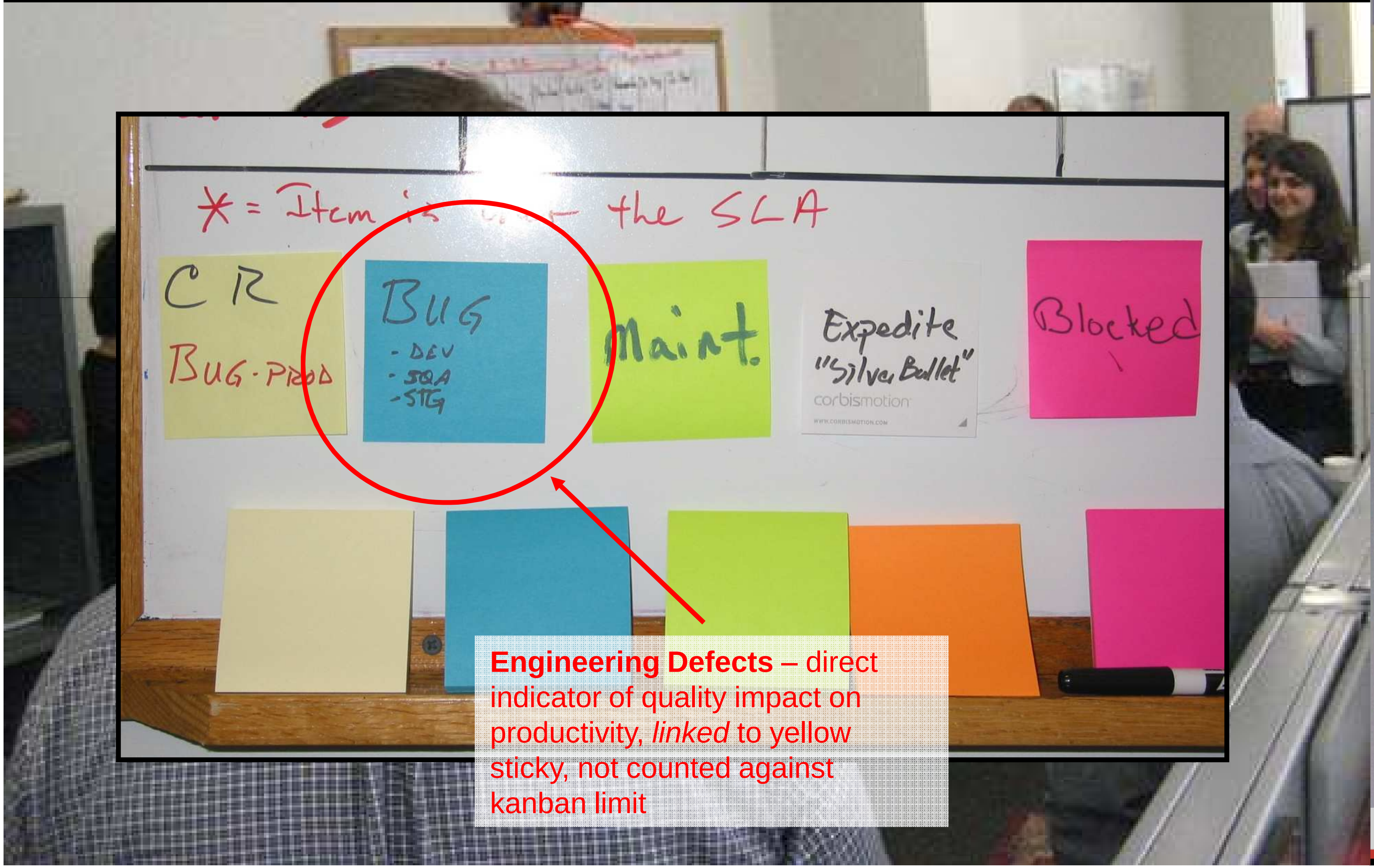
Flow – from Engineering Ready to Release Ready

Colors are used to designate classes of service for work items



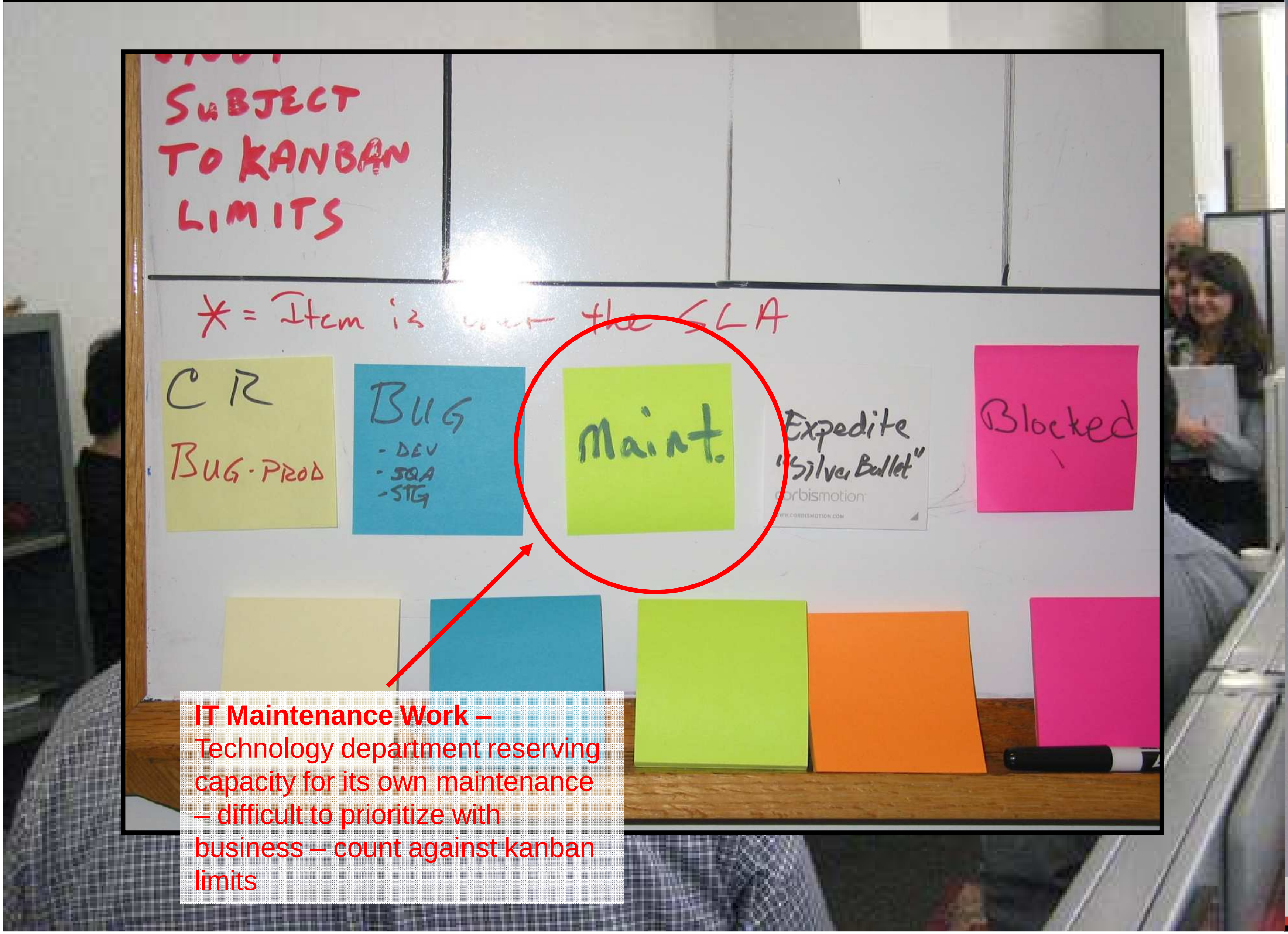
Change Requests and Production Bugs – Customer valued and prioritized by governing board

Quantity of blue tickets on the board is an immediate indicator of development quality that is impeding flow of customer valued work and reducing throughput



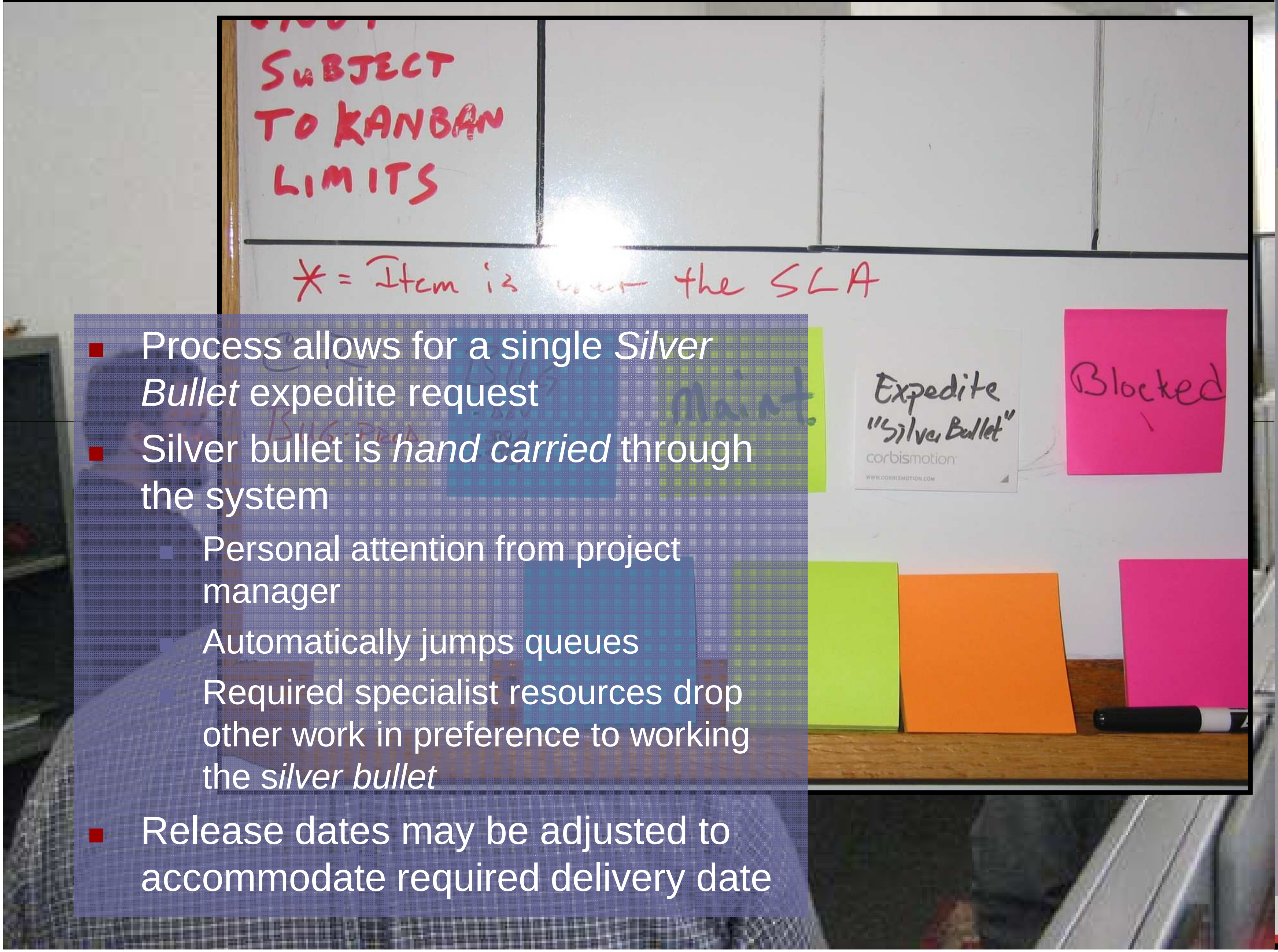
Engineering Defects – direct indicator of quality impact on productivity, *linked* to yellow sticky, not counted against kanban limit

Non-customer valued but essential work is tracked as a different class of work



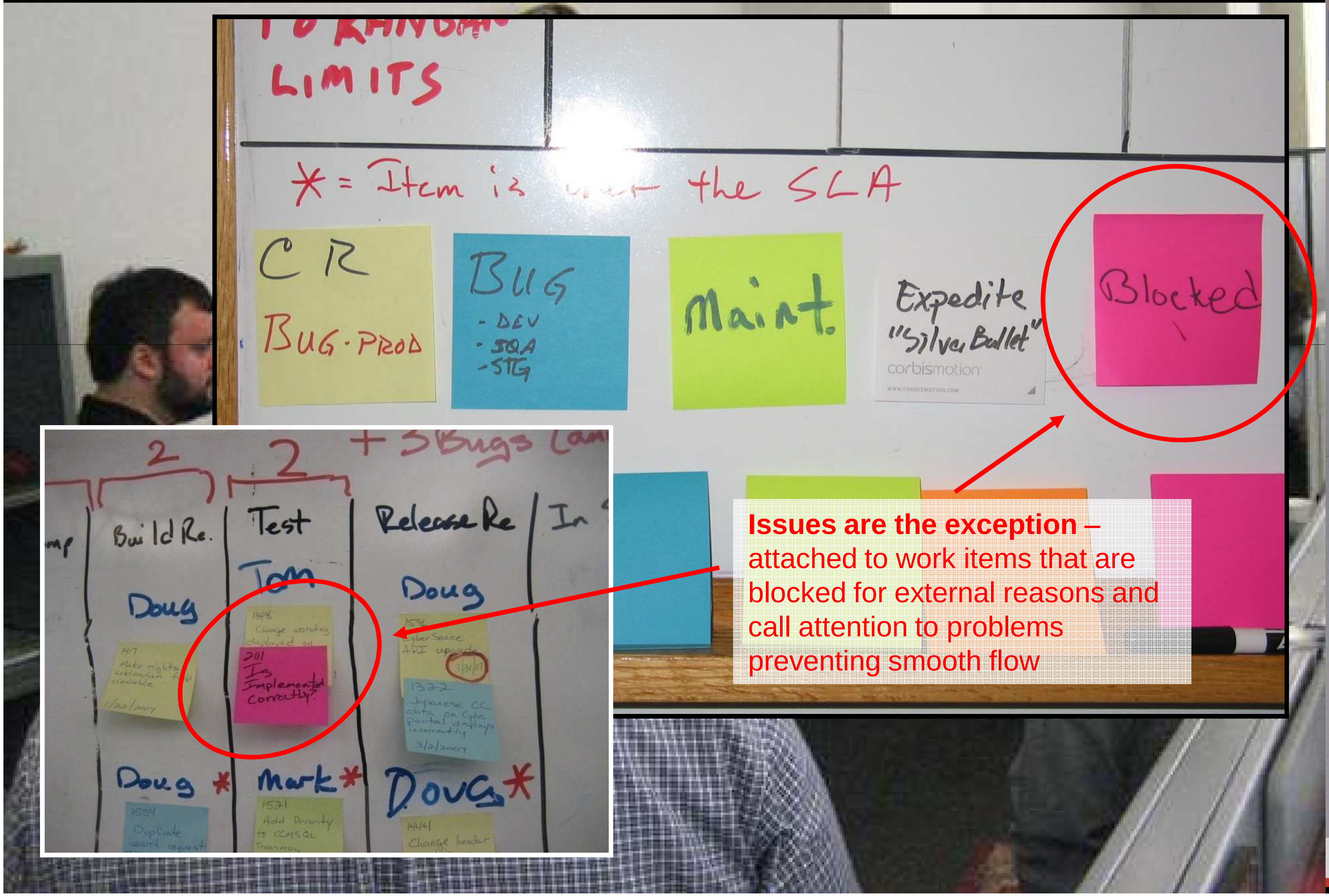
IT Maintenance Work –
Technology department reserving capacity for its own maintenance – difficult to prioritize with business – count against kanban limits

Expediting – *the Silver Bullet*



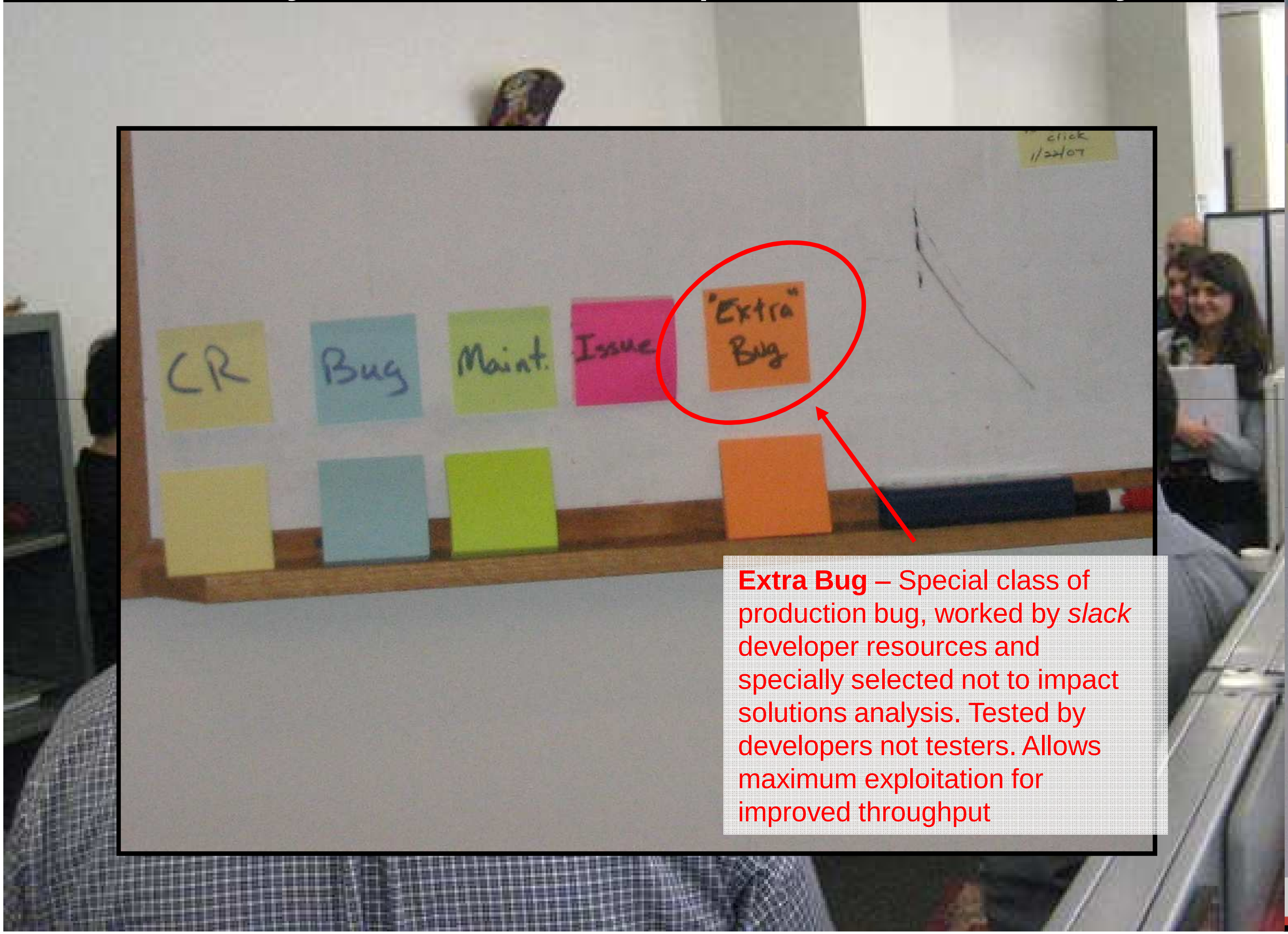
- Process allows for a single *Silver Bullet* expedite request
- Silver bullet is *hand carried* through the system
  - Personal attention from project manager
  - Automatically jumps queues
  - Required specialist resources drop other work in preference to working the *silver bullet*
- Release dates may be adjusted to accommodate required delivery date

Quantity of pink *issue* tickets on board directly indicates flow impacting problems that need attention from management



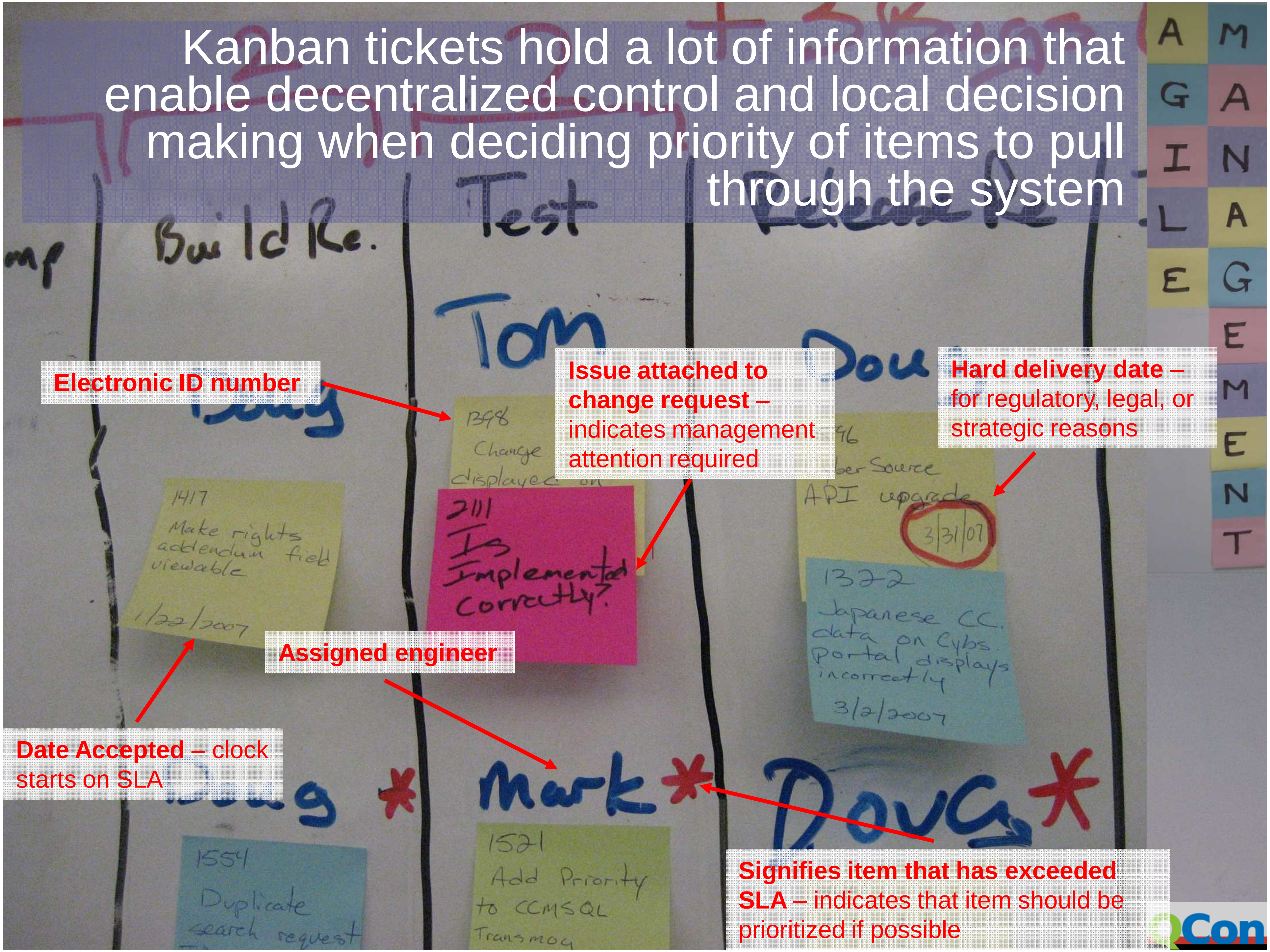
Issues are the exception – attached to work items that are blocked for external reasons and call attention to problems preventing smooth flow

Temporary classes of work may be introduced tactically to maximize exploitation of the system



Extra Bug – Special class of production bug, worked by *slack* developer resources and specially selected not to impact solutions analysis. Tested by developers not testers. Allows maximum exploitation for improved throughput

Kanban tickets hold a lot of information that enable decentralized control and local decision making when deciding priority of items to pull through the system



Electronic ID number

Issue attached to change request – indicates management attention required

Hard delivery date – for regulatory, legal, or strategic reasons

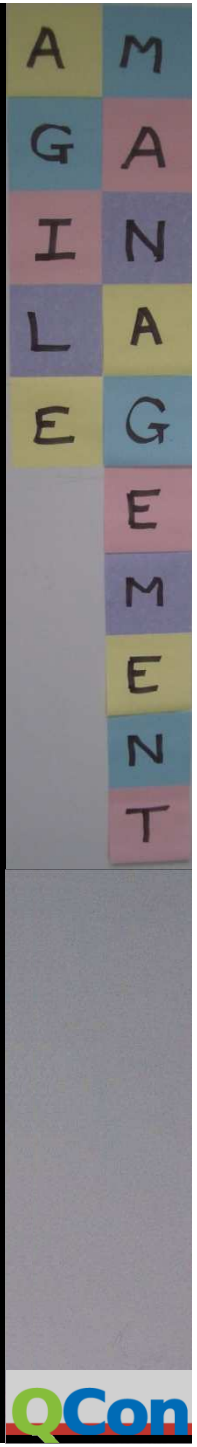
Assigned engineer

Date Accepted – clock starts on SLA

Signifies item that has exceeded SLA – indicates that item should be prioritized if possible

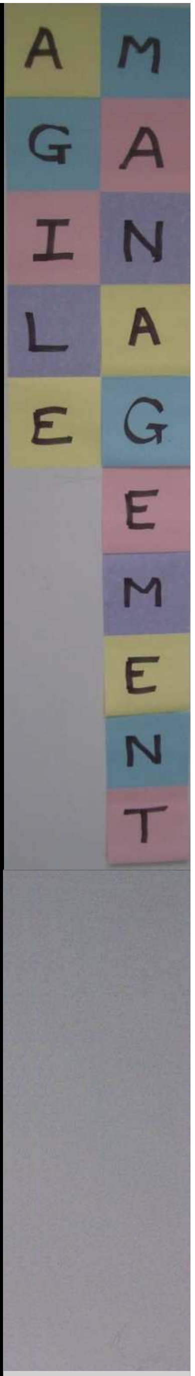
Kanban delivers *iterationless* development

- Releases were agreed and planned for every 2nd Wednesday
- Prioritization Board meetings were held every Monday
- Release content is bound and published only 5 days prior
- Prioritization meetings are required only to answer the question, “Which items from the backlog do we want to select this week to fill any empty slots in the input queue?”
- Prioritization holds change request selection until the *last responsible moment*
- It keeps (real) options open

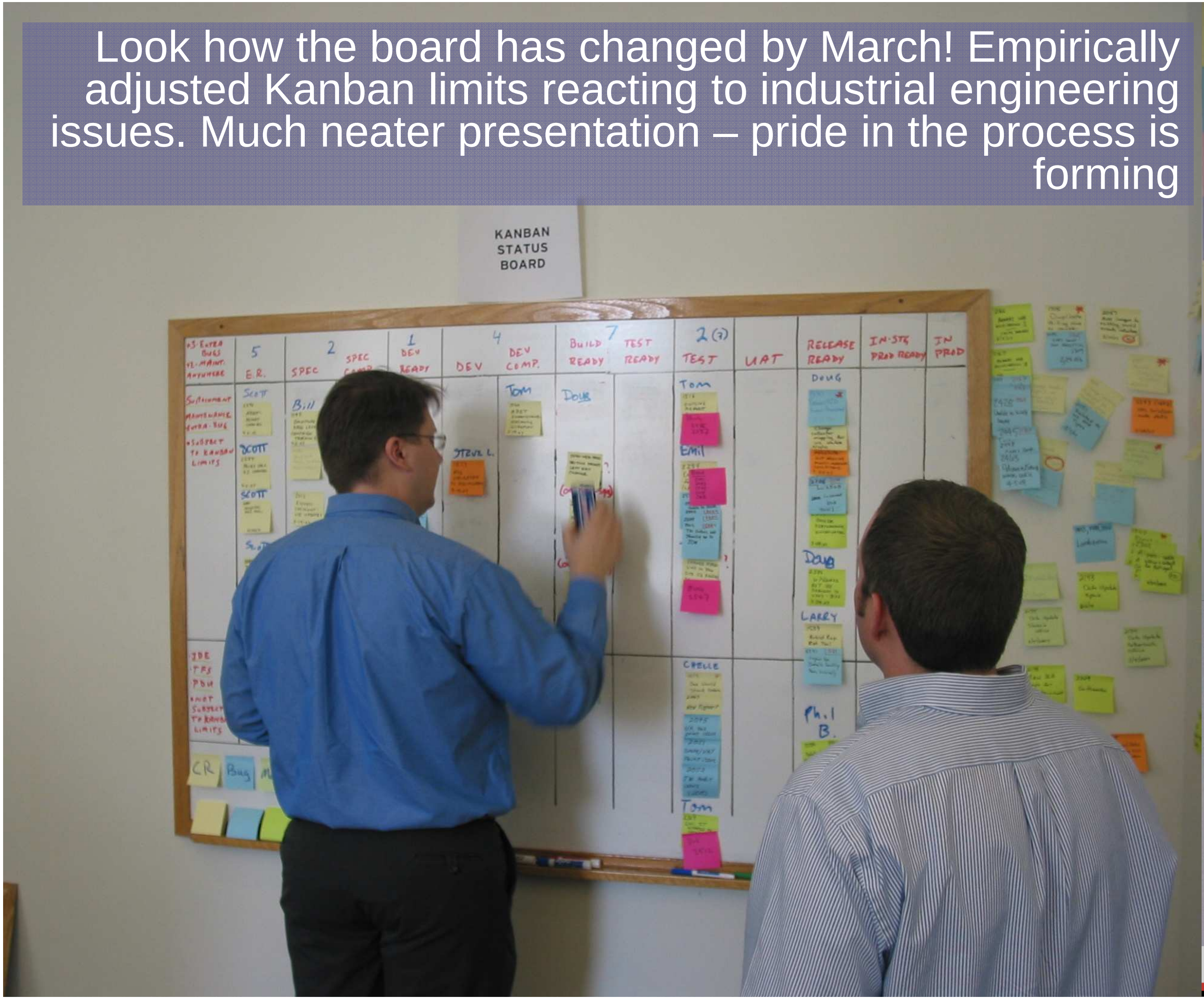


Kanban innovates on typical agile/iterative development by introducing a late binding release commitment

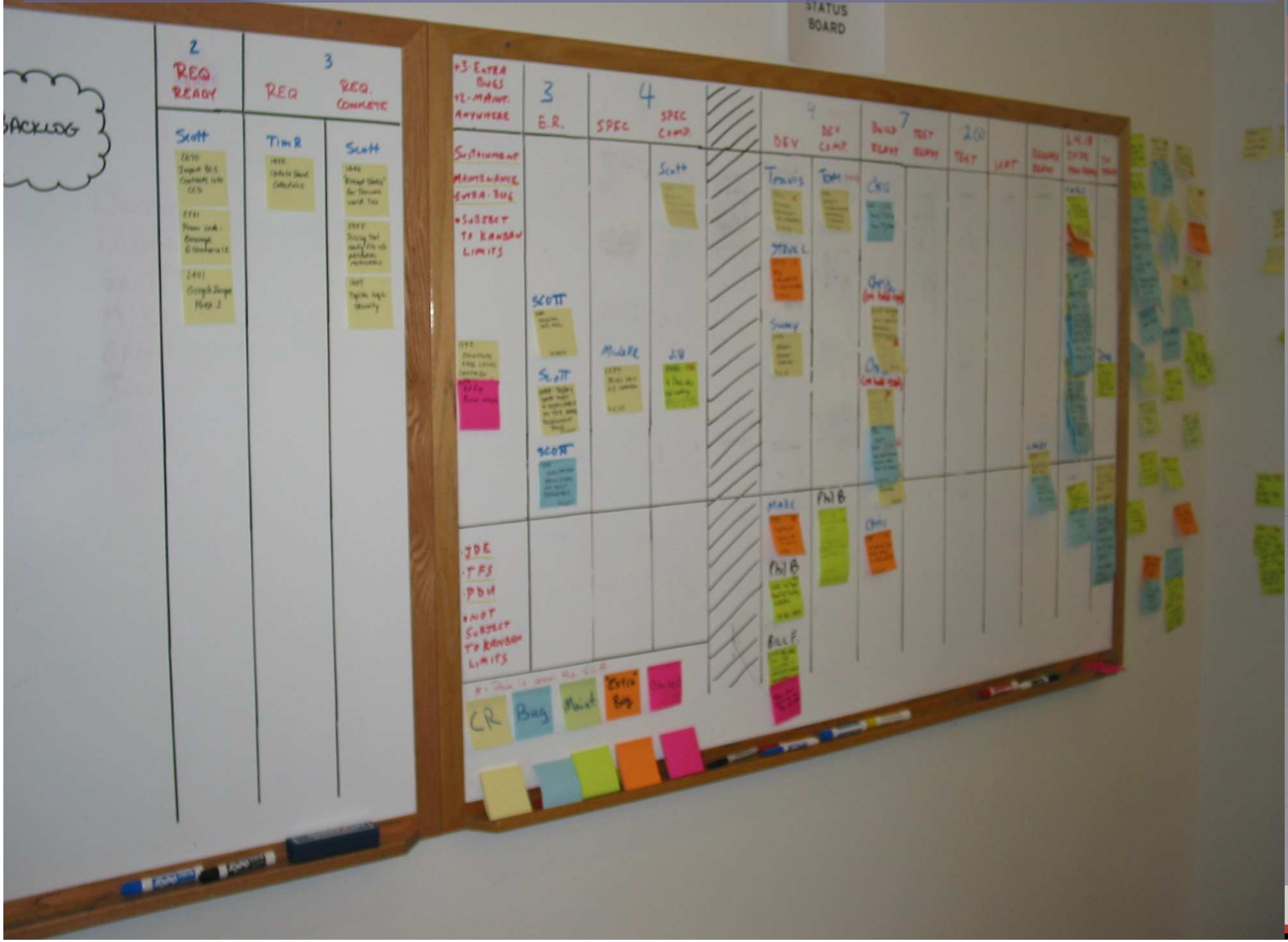
- Kanban system breaks constraint of typical agile/iterative 2-4 week cycle
- Requests can take up to 100 days to process but releases still made every 14 days
- Average item takes 14 days of engineering
- Input and sizing is decoupled from cadence of releases
- Decision on content of release made 5 days prior to release
- No estimation is done on individual items
- Effort to estimate is turned back to productivity (analysis, coding, testing)



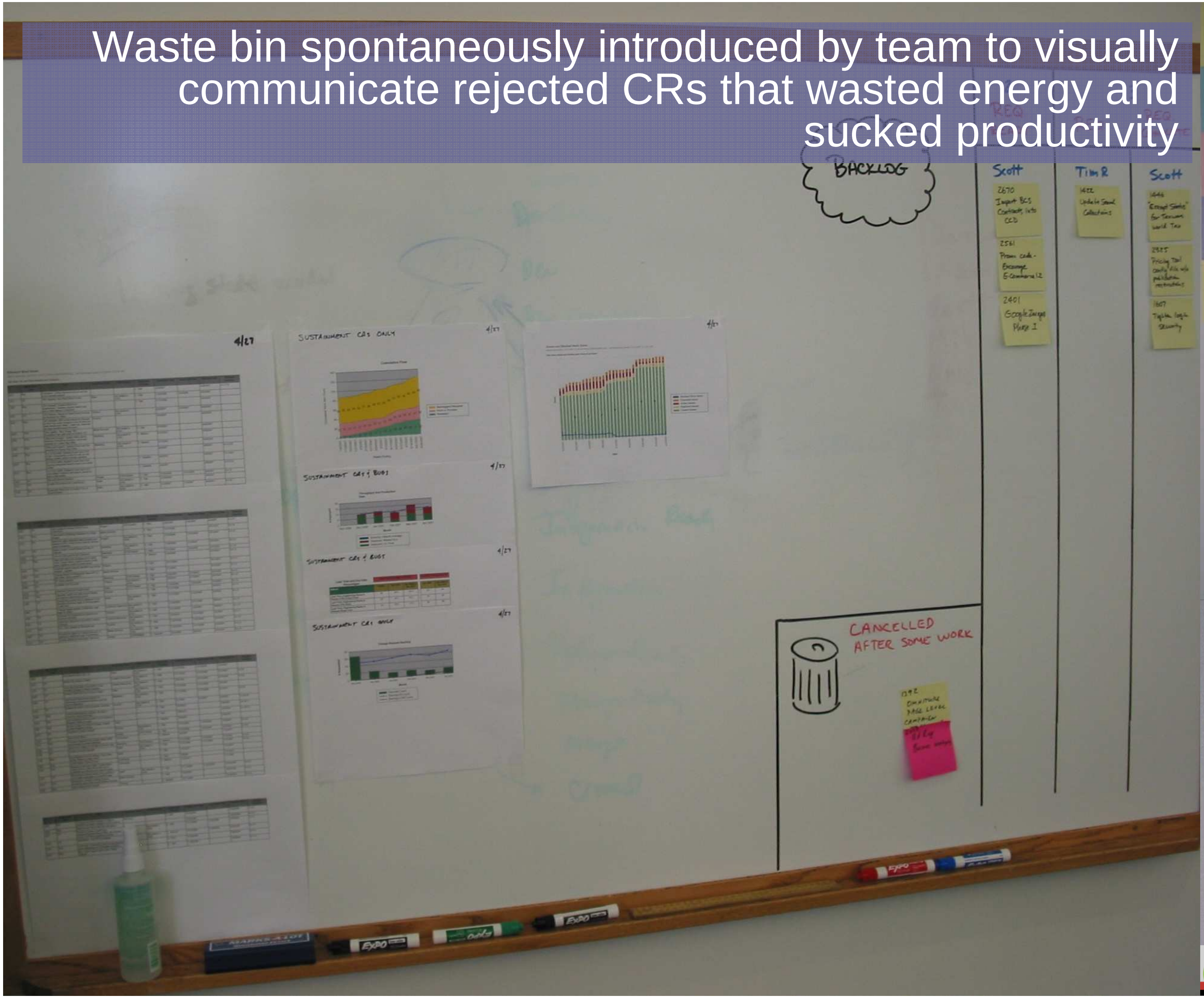
Look how the board has changed by March! Empirically adjusted Kanban limits reacting to industrial engineering issues. Much neater presentation – pride in the process is forming



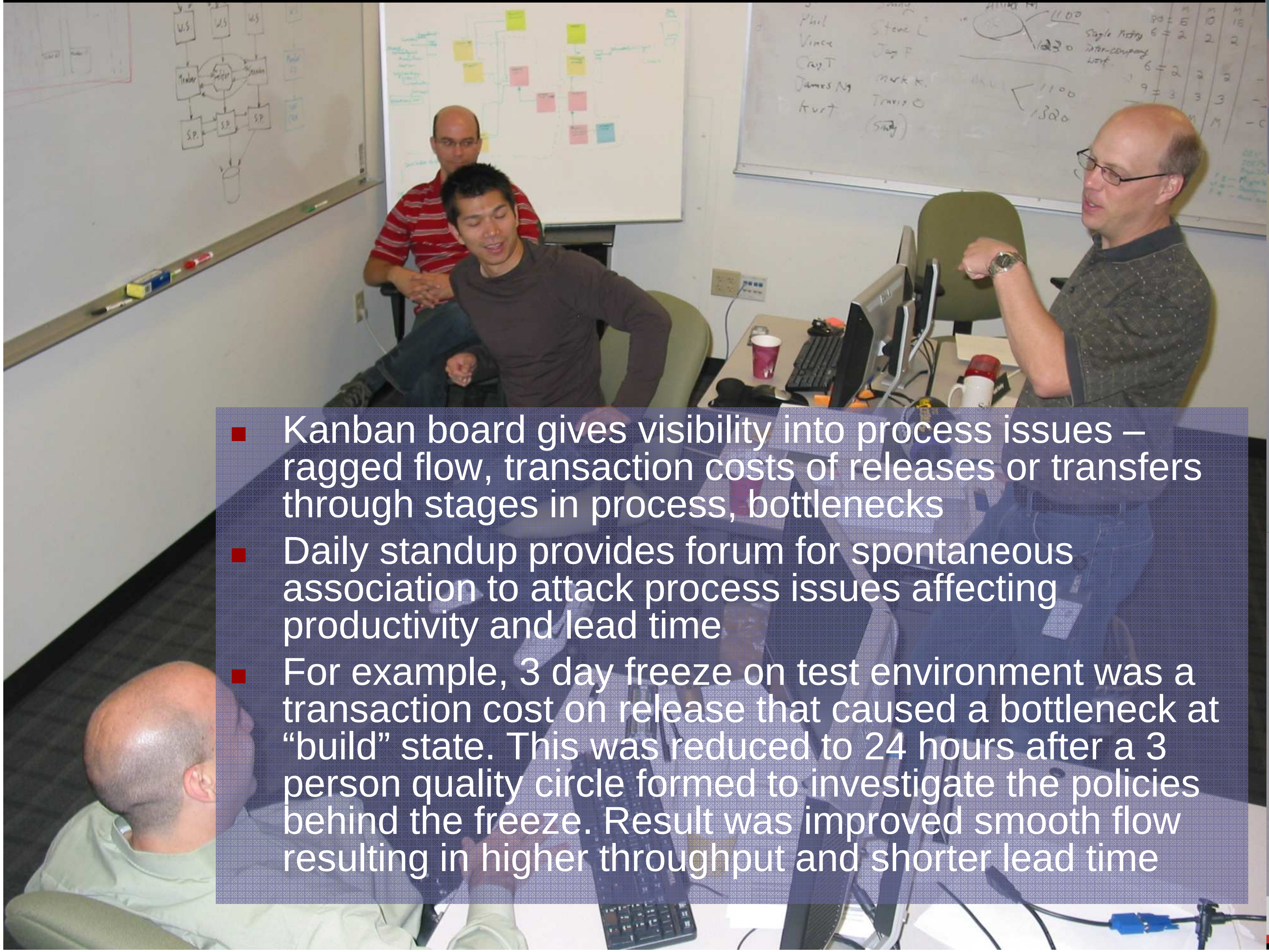
And again in April, more changes to Kanban limits and forward extension of the process to business analysis



Waste bin spontaneously introduced by team to visually communicate rejected CRs that wasted energy and sucked productivity

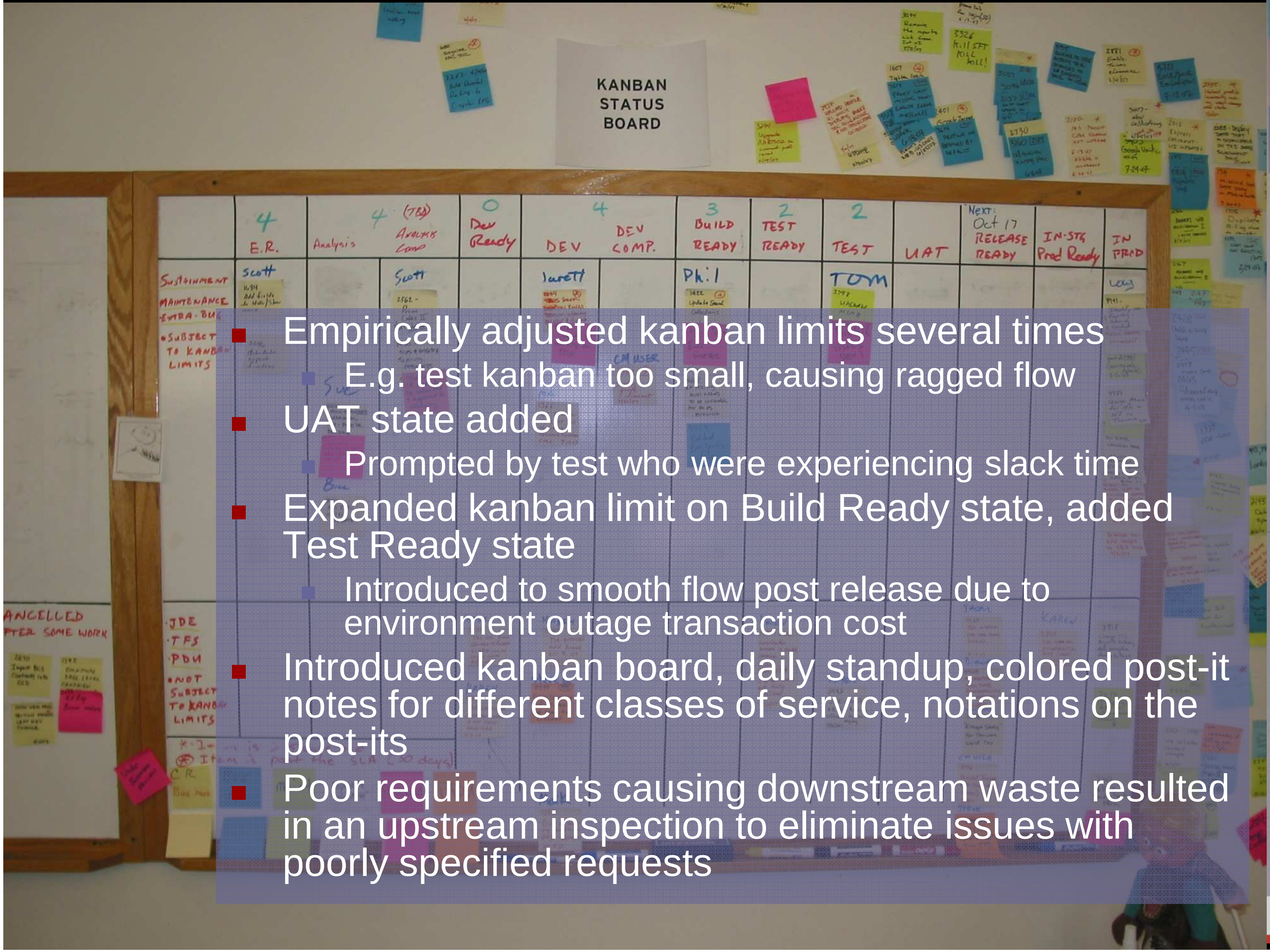


Spontaneous Quality Circles started forming



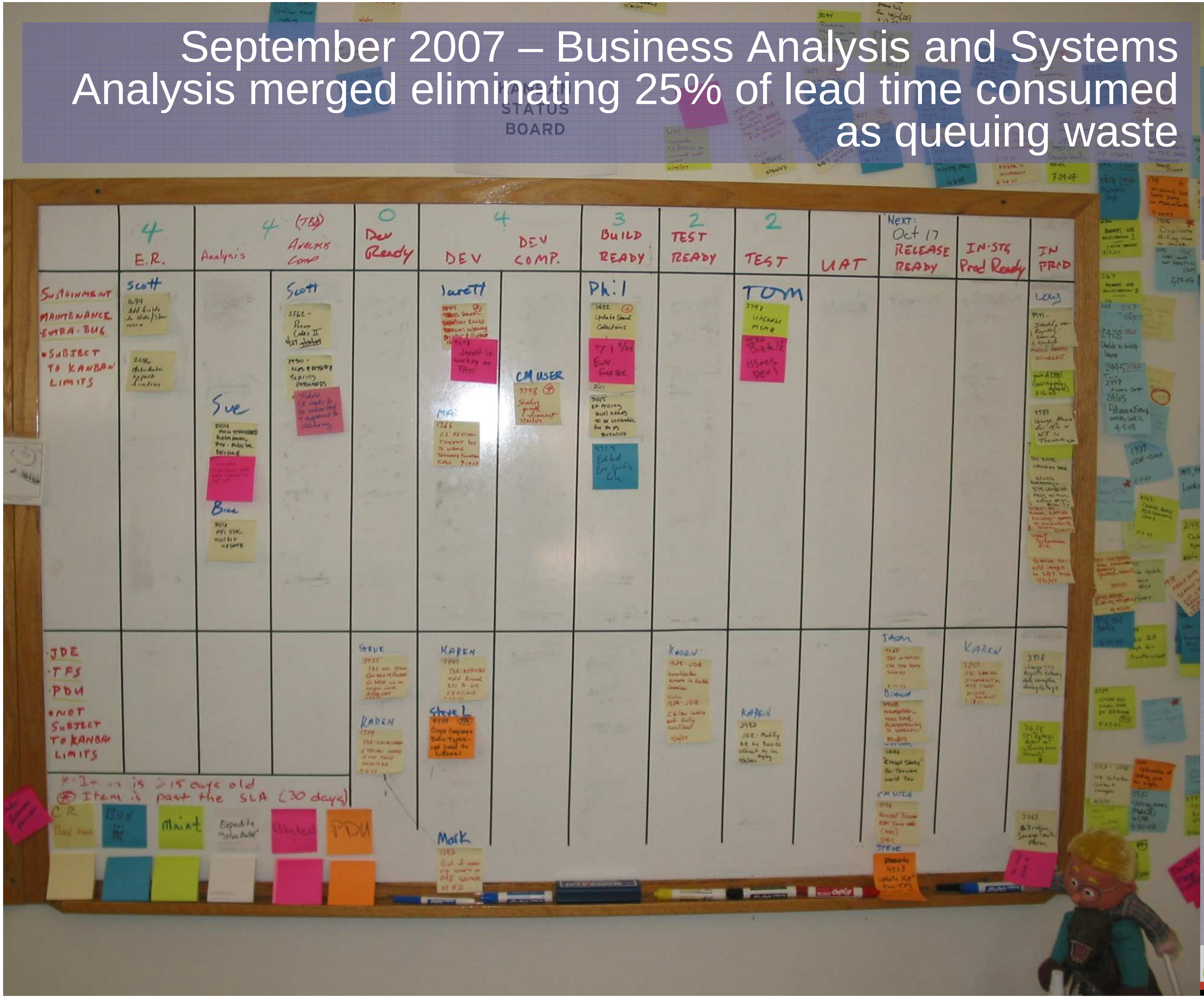
- Kanban board gives visibility into process issues – ragged flow, transaction costs of releases or transfers through stages in process, bottlenecks
- Daily standup provides forum for spontaneous association to attack process issues affecting productivity and lead time
- For example, 3 day freeze on test environment was a transaction cost on release that caused a bottleneck at “build” state. This was reduced to 24 hours after a 3 person quality circle formed to investigate the policies behind the freeze. Result was improved smooth flow resulting in higher throughput and shorter lead time

Other spontaneous quality circle kaizen events

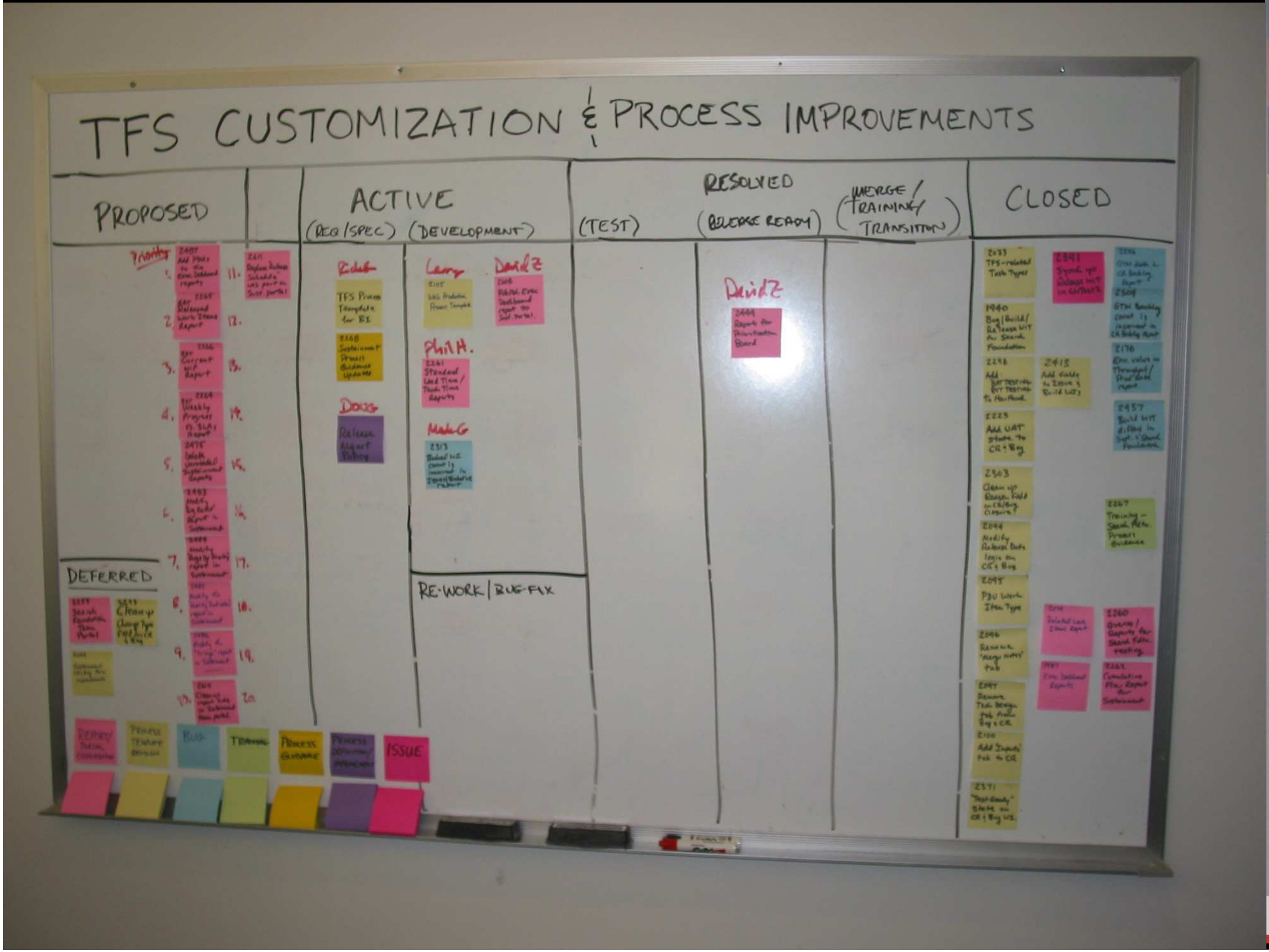


- Empirically adjusted kanban limits several times
  - E.g. test kanban too small, causing ragged flow
- UAT state added
  - Prompted by test who were experiencing slack time
- Expanded kanban limit on Build Ready state, added Test Ready state
  - Introduced to smooth flow post release due to environment outage transaction cost
- Introduced kanban board, daily standup, colored post-it notes for different classes of service, notations on the post-its
- Poor requirements causing downstream waste resulted in an upstream inspection to eliminate issues with poorly specified requests

September 2007 – Business Analysis and Systems Analysis merged eliminating 25% of lead time consumed as queuing waste



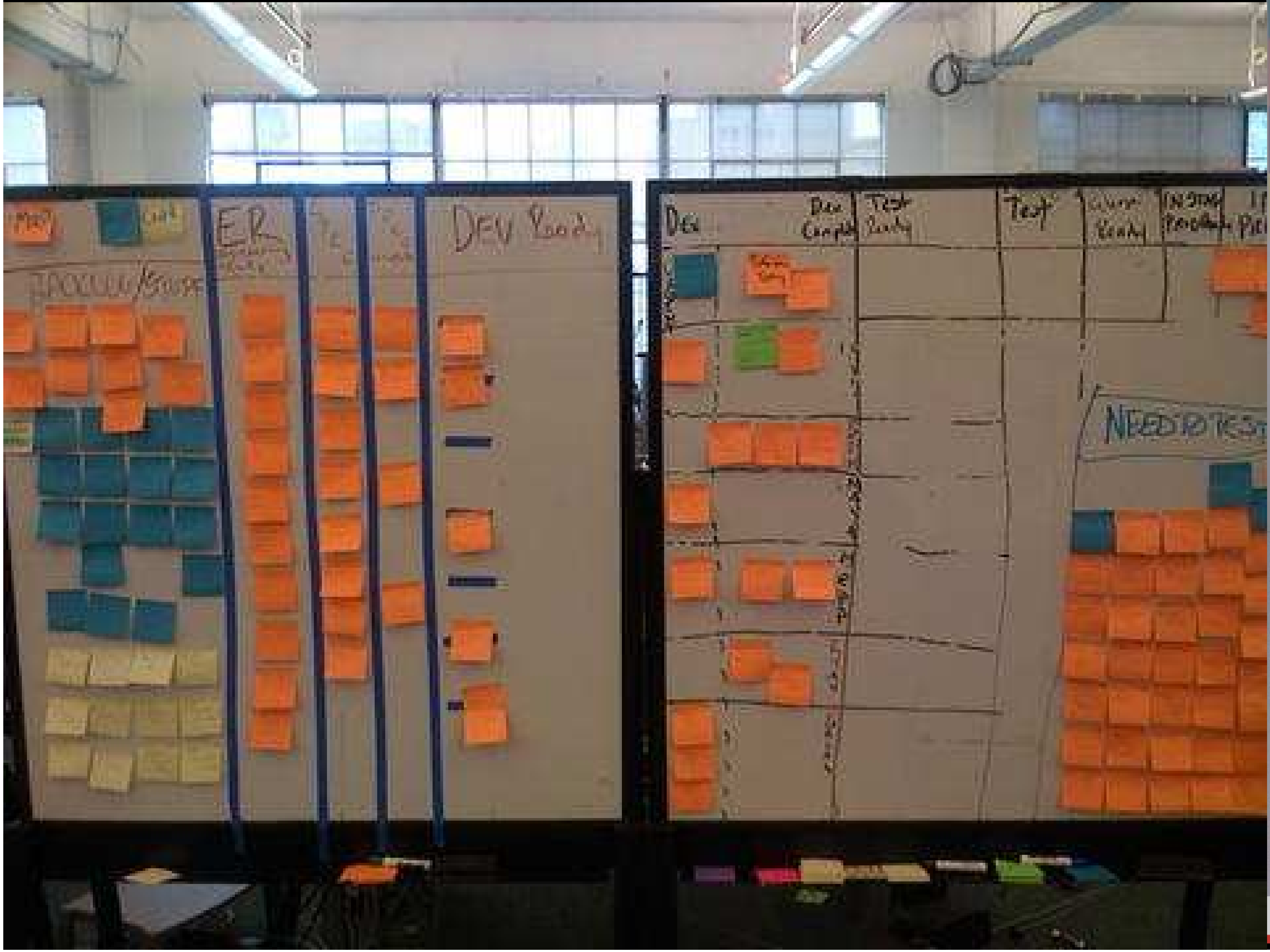
And the process is spreading inside Corbis …



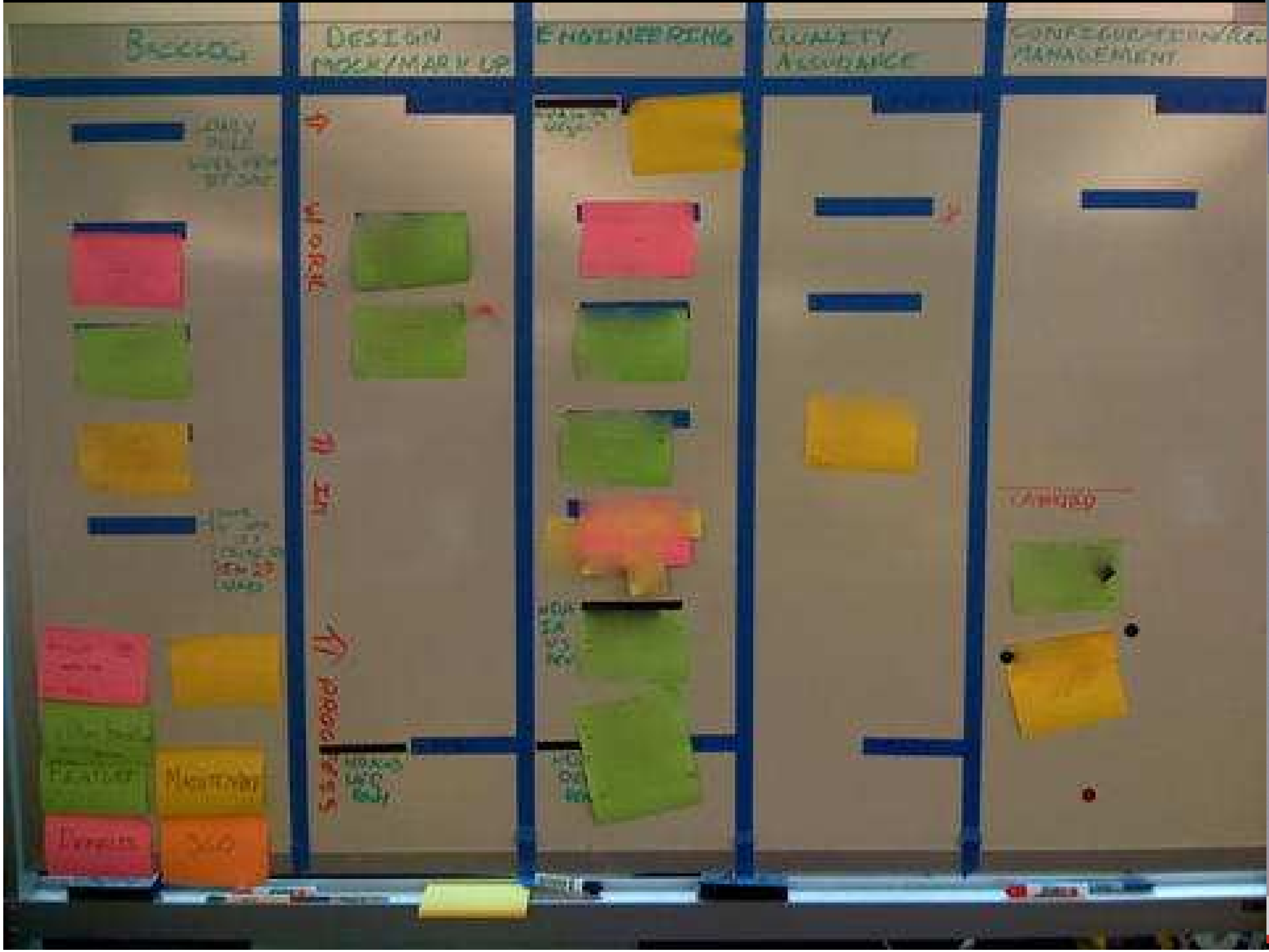
AGILE MANAGEMENT

QCon

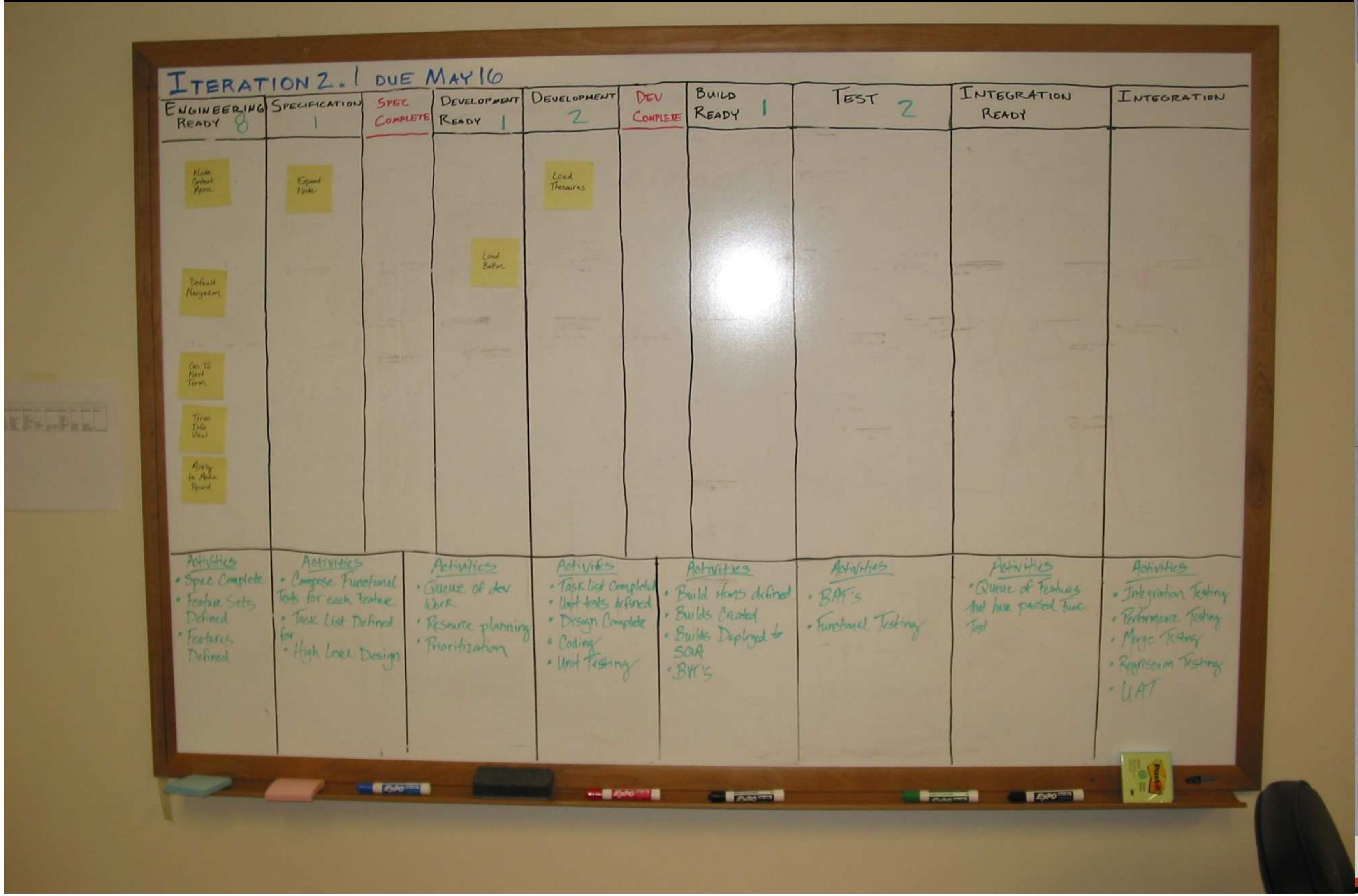
And externally at companies like Yahoo! …



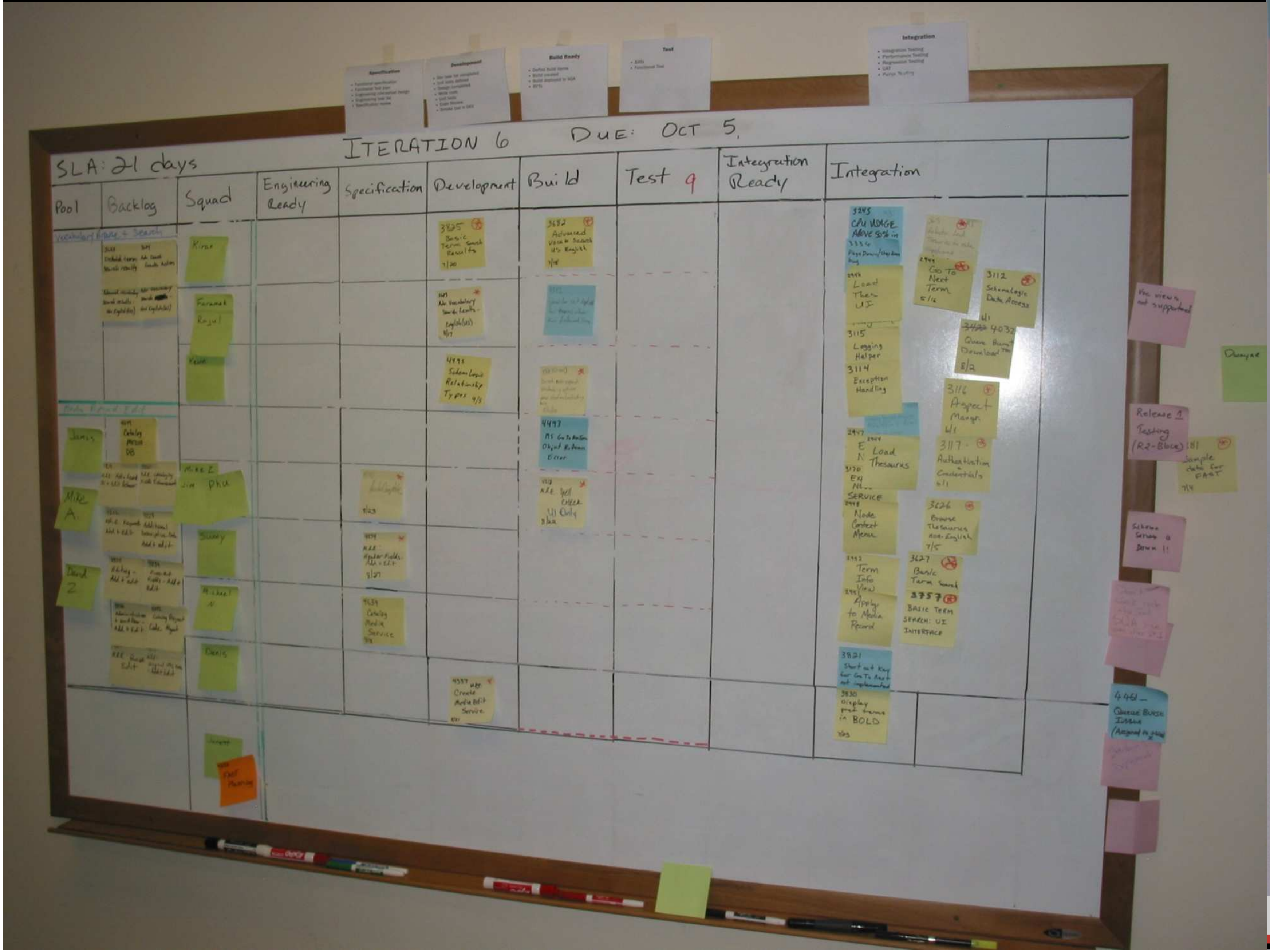
… this one on the Mash social network team



And the technique is being introduced to major projects with much longer time horizons. This example has a monthly “integration event” rather than a release every two weeks

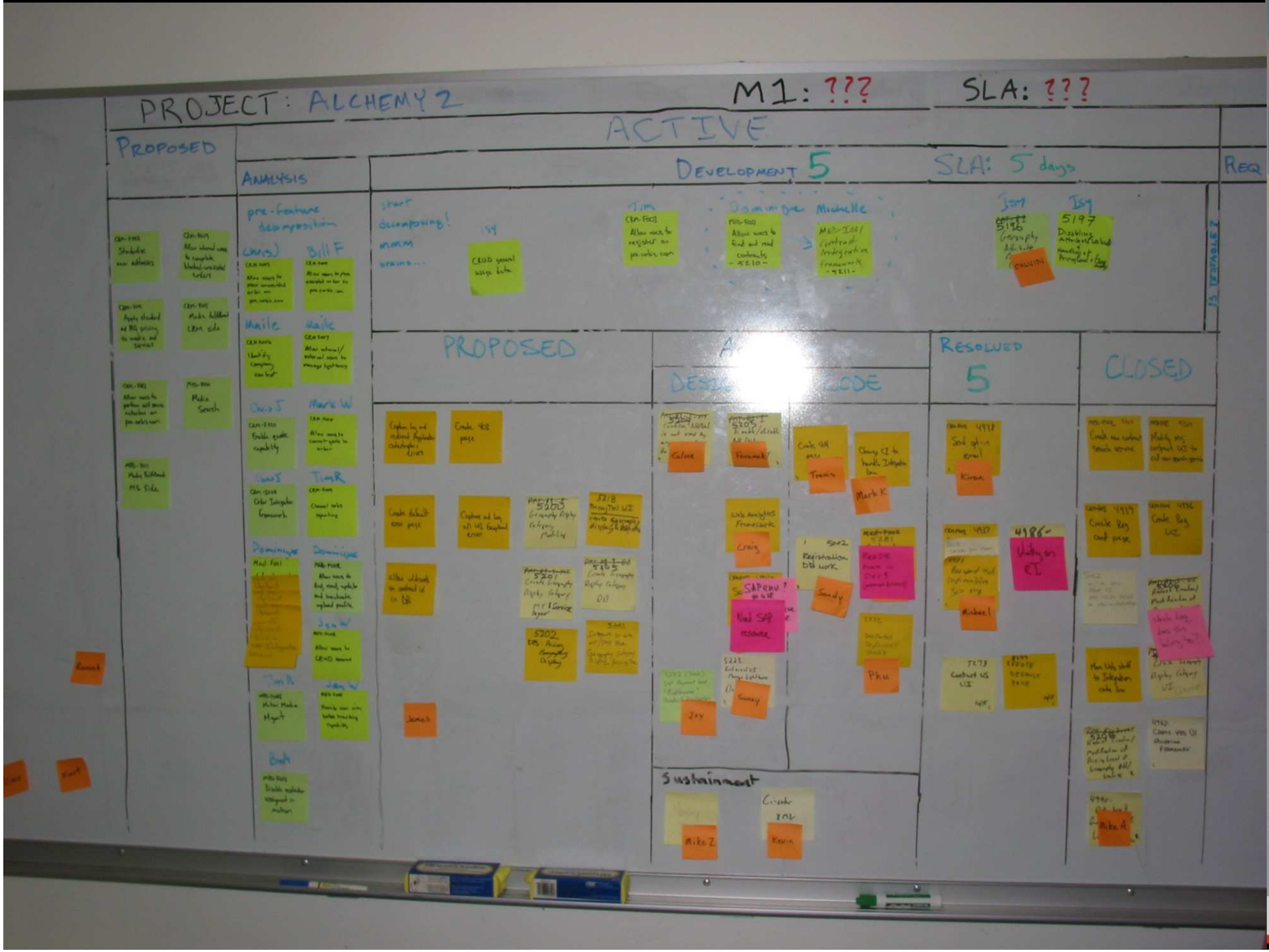


5 months later significant changes are evident

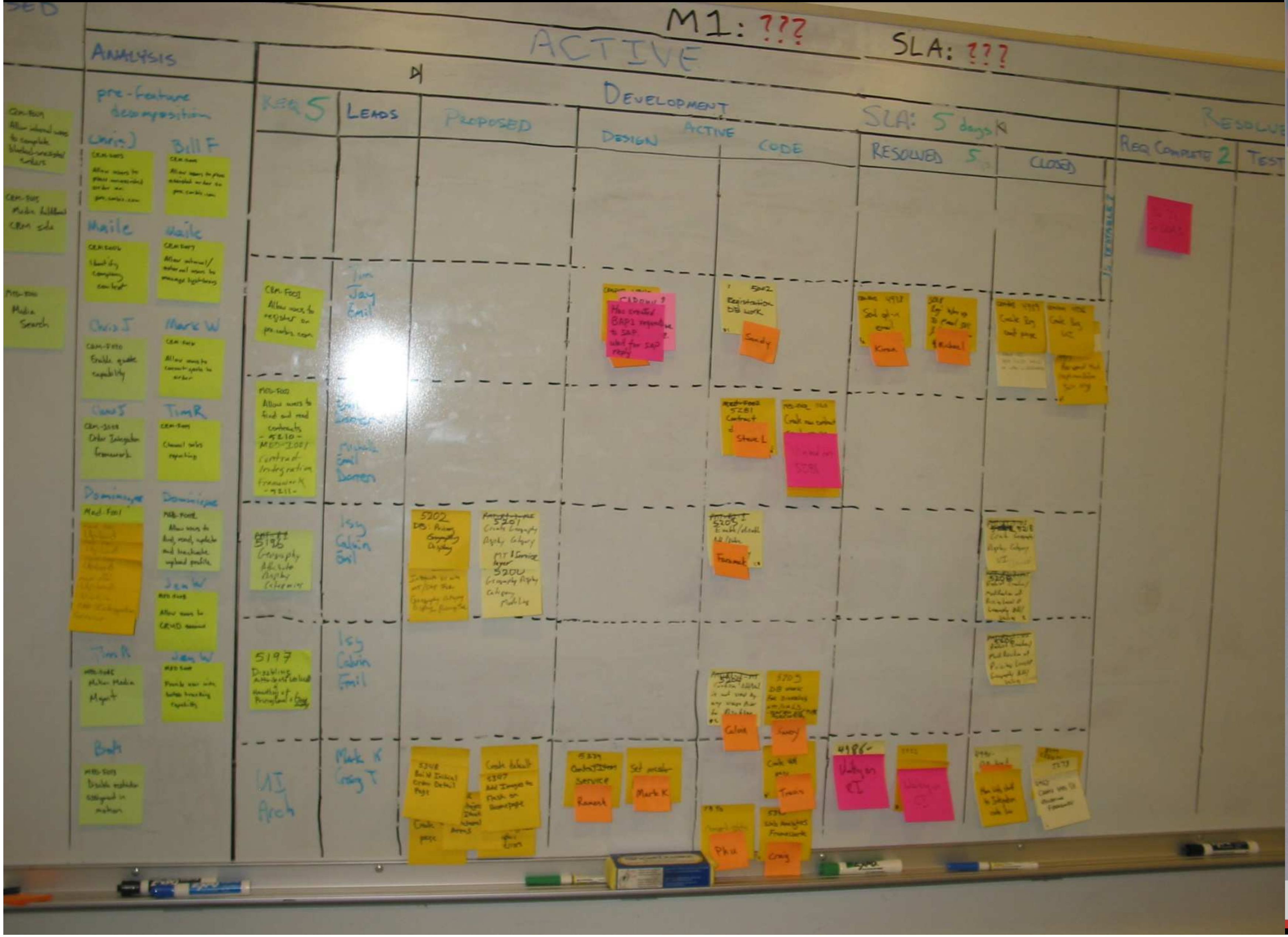


AGILE MANAGEMENT

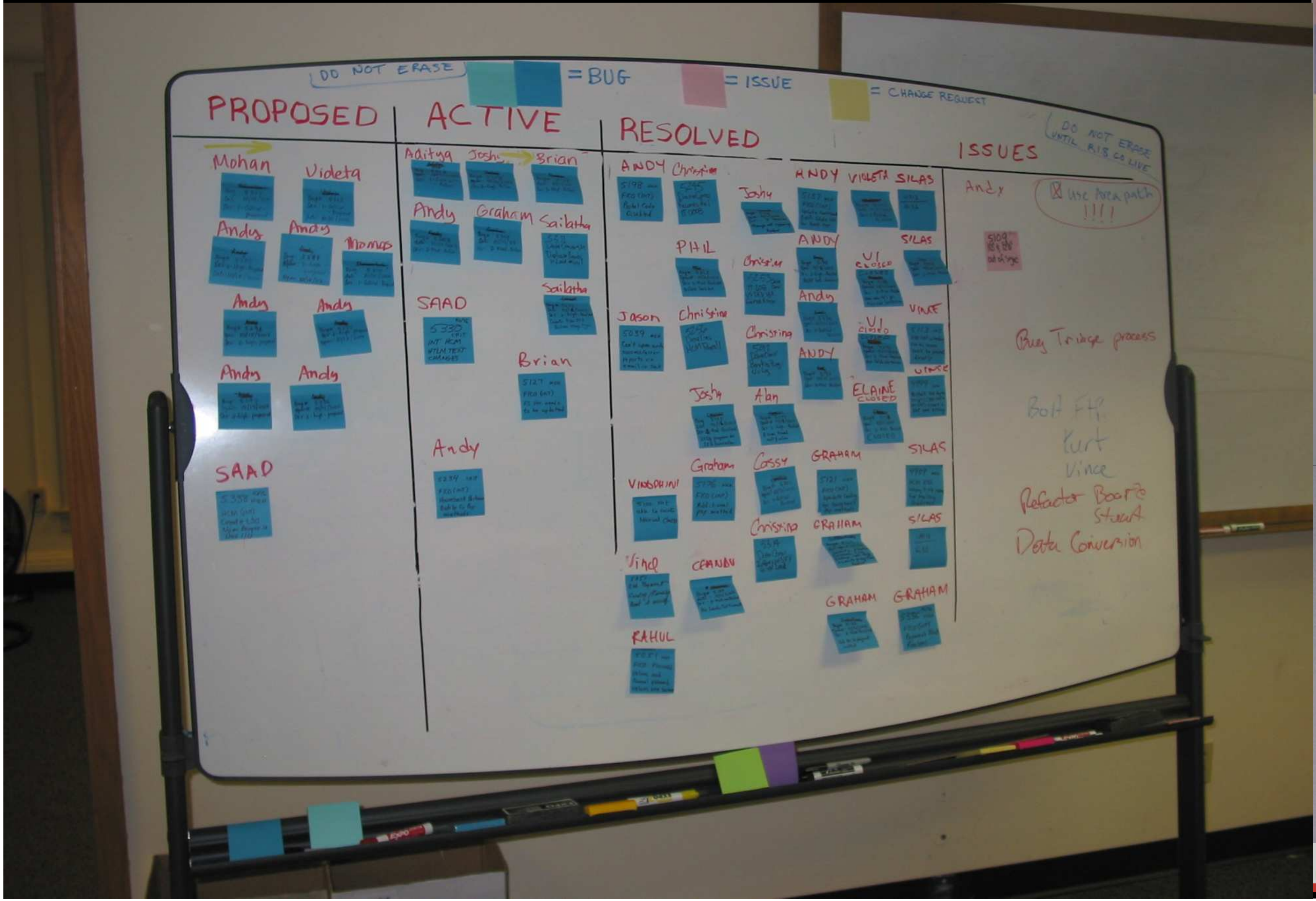
Major project with two-tiered kanban board



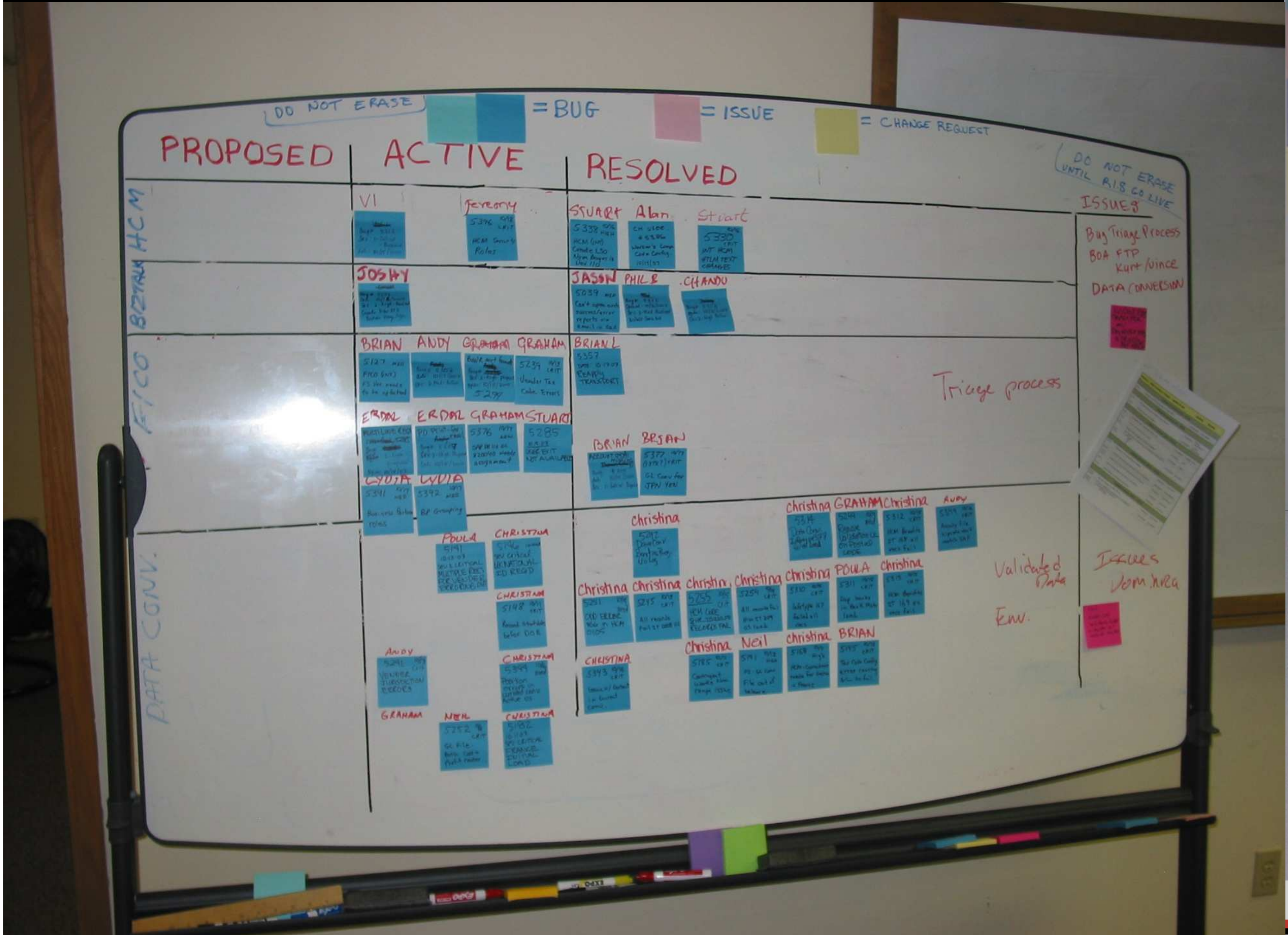
Major Project with two-tiered kanban board using swim lanes for *feature sets*



Less mature major project in trouble adopts kanban to bring a focus to daily routine and visibility to work-in-progress to team and management



Next Day – Team is maturing quickly and has refactored the board with swim lanes for functional areas

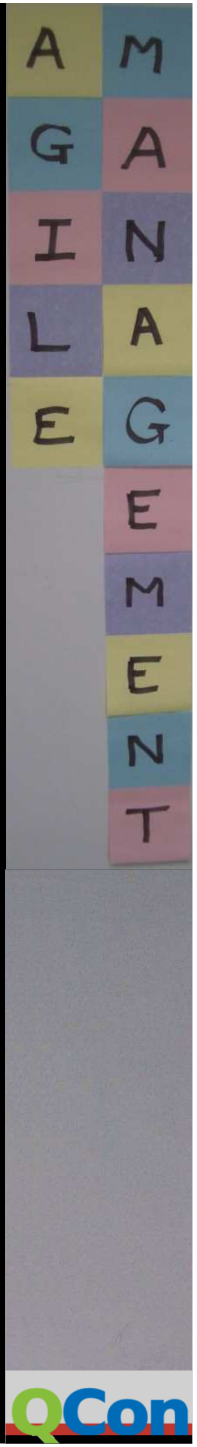


Kanban has allowed scaling standup meetings to much larger teams than is typical with Scrum

In this example more than 40 people attend a standup for a large project with 6 concurrent development teams. The meeting is usually completed in approximately 10 minutes. Never more than 15.

Bargaining, Democracy & Collaboration

- First 8 weeks prioritization board would bargain against the available slots and WIP limit
  - I've got two small requests can you treat them as one?
- People started to lobby each other and build business cases to get items selected
- Familiarity with the system led to the consensus decision to adopt a democratic process
- 3 months later it was evident that democracy didn't always select the best candidate
- And it was replaced with a collaborative process based on strategic and current tactical marketing objectives

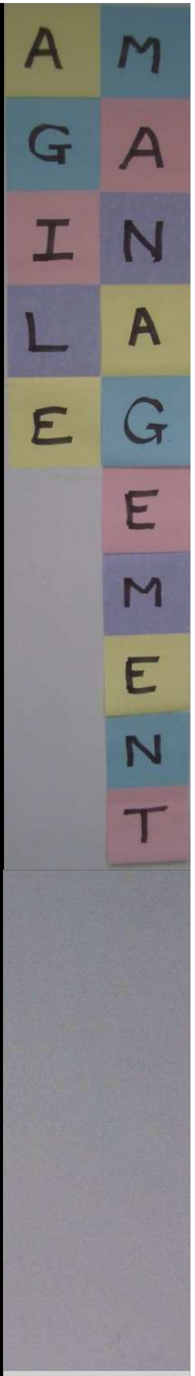


The process has shown remarkable robustness to gaming from the business

- Prioritization board consists of VPs from 6 business units
- Understanding that expediting costs throughput and lead time has resulted in an expectation that only critical items qualify for *Silver Bullet* status
- Attempts to game prioritization by setting a delivery date are tightly scrutinized by the board
- As a result the process is self-regulating with the prioritization board enforcing the anti-gaming rules
- As a result the *Silver Bullet* and delivery date options are seldom used

Summary

- Culture Change
  - Trust, empowerment, objective data measurement, collaborative team working and focus on quality
- Policy Changes
  - Late-binding release scope, no estimating, late-binding prioritization
- Regular delivery cadence
- Cross-functional collaboration
  - Previously unheard of VP level selfless collaboration on business priority
- Self-regulating process robust to gaming and abuse
- Continuous Improvement
  - Increased throughput, high quality, process continually evolving, kanban limits empirically adjusted



QCon

Learn More

- Join the Kanbandev Yahoo! Group
- Corey Ladas' Lean Software Eng Blog
  - http://leansoftwareengineering.com/
- Agile Management Blog
  - http://www.agilemanagement.net/Articles/Weblog/blog.html

Thank you!

dja@agilemanagement.net

http://www.agilemanagement.net/

About…

David Anderson is a thought leader in managing effective software teams. He is the President of Modus Cooperandi, a consulting firm dedicated to improving leadership in the IT and software development sectors.

He has 25 years experience in the software development business starting with computer games in the early 1980’s. As a pioneer in the agile software movement David has managed teams at Sprint, Motorola and Corbis delivering superior productivity and quality. At Microsoft he developed the MSF for CMMI Process Improvement methodology.

David’s book**, Agile Management for Software Engineering** – *Applying the Theory of Constraints for Business Results*, introduced many ideas from Lean and Theory of Constraints in to software engineering.

David was a founder and is a current board member of the APLN, a not for profit dedicated to promoting better standards of leadership and management in knowledge worker industries.
He can be contacted at…
Email: dja@moduscooperandi.com