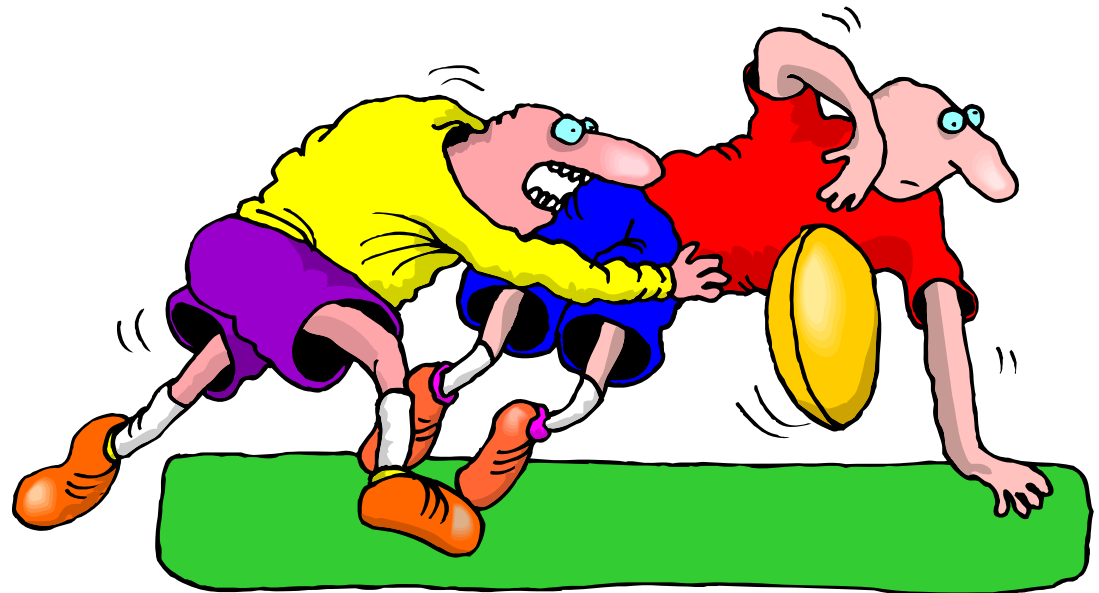


# How NOT to Do Scrum

Patterns and Anti-patterns



New York City Scrum User Group June 17, 2010

# Speaker Bio

**Andrew Kazarinoff CSP / CSM**  
Qualytic Consulting, New York

**ScrumMaster, Agile Coach, Trainer**

**Two Big Banks**

**Financial Publisher**

**System Integrator**

**CRM Services Company**

**Storage Systems Company**

**Background:** PMO-based process improvement  
Traditional project management

**[Andrew.Kazarinoff@qualytic.com](mailto:Andrew.Kazarinoff@qualytic.com)**

# What this is about

## Patterns

Practices that have proved to be generally effective

## Anti-Patterns

Coplien: an anti-pattern is something that looks like a good idea but which backfires badly when applied

Schwaber: what used to make you feel safe

**While Anti-Patterns may not make your Scrum project fail, we value the Patterns more**

**Context Determines**

**(Not a catalog of failure modes, Scrum Smells, ScrumButs)**

Announce: “We’re going Agile. All teams will do Scrum”

- First demonstrate benefits with a pilot project
- Cultivate and support buzz / viral spread



Transfer project managers into ScrumMaster roles

Send project managers to a formal CSM course, then

- 1) train with teams
- 2) they shadow you
- 3) you mentor them

Remove cubicle partitions, move into a team room

Let the teams decide

Let an available product manager or marketing executive be the Product Owner

1) Carefully screen 2) send to formal CSPO course 3) mentor

Assume there will be a Vision and Product Roadmap

Begin immediately to work with the Product Owner

Engage BAs

Handle blockages as they come up

Begin immediately to identify and eliminate waste, enterprise impediments

- non-dedicated resources (time-slicing), unclear support roles, deployment delays
- map the value stream



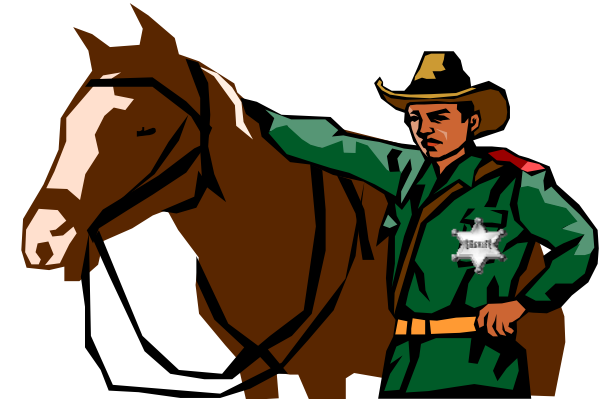
# TRAINING

Train everybody immediately

Observe first, talk – no “new sheriff in town”

If Scrum has been in place, a deep-dive retrospective

- hooks for micro-training on planning, story estimating, practices for high productivity



Give the Product Owner special training

Train POs (CSPOs) with the teams they will join, then mentor

# SPRINT PLANNING

(1)

Trust that the Product Owner is doing real capability and release planning

Assume the Product Owner will key user stories to capabilities and vision

Buyer can display categories

Buyer can display brands

Seller can enter descriptions

Item List  
Search  
Bidding



Assume the Product Owner will do continuous backlog grooming

Keep after and coach the Product Owner

Engage BAs, whole team

(READY)

# Must-Have for High Productivity -- READY

## READY

- Product backlog prioritized on business value
- Team understands the release plan / roadmap
  - Stories clearly keyed to target capabilities
- Stories understood by developers and testers
  - Story Syntax Observed
- Acceptance tests designed



Sutherland

# SPRINT PLANNING

(2)

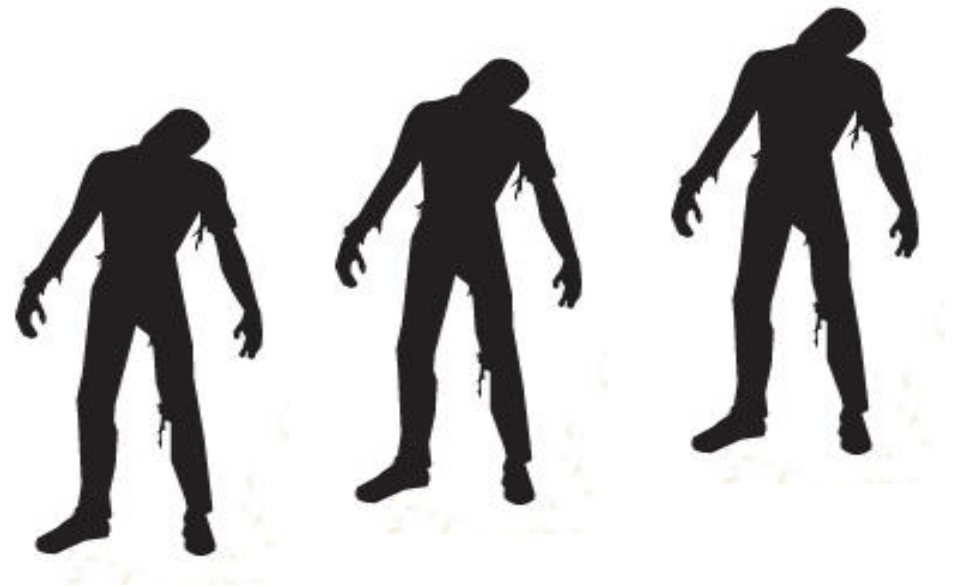
The PO e-mails a list of user stories

The Tech Lead decomposes the stories into tasks

The SM assigns “story owners”

The Tech Lead estimates the tasks

The Tech Lead assigns tasks



(READY)

Allow the team to self-organize

# Must-Have for High Productivity – Self-Organization

## **SELF-ORGANIZATION** (“Secret Sauce”)

“Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done”

- Team decides how to achieve goals
- Collaborates rather than following instructions
- Reports to itself, not to the PO or SM



Sutherland

# COMMUNICATION

In daily scrums, let the team members report to the ScrumMaster or Product Owner

(SELF-ORGANIZATION)

Let the team sit at their desks for daily scrums

Physical stand-ups even with distributed teams

Skip the daily scrum if the team is really busy

Sutherland warns against tampering with the framework

Have a status meeting every week

(SELF-ORGANIZATION)

# SPRINTS

Team members report task progress hours

(self-organization, but the ALM uses)

Vary the sprint length to fit the work planned

Don't destroy the cadence and disorient the team

Keep an open door to management, sales, and PO

(Nokia Test: There are no project managers -- or anyone else -- disrupting the work of the team)

Feed them information – sprint goal, progress on capabilities

Alternate “QA” test/fix sprints with coding sprints

Defect Cost Increase after 3 weeks = 24 x

(DONE)

# Must-Have for High Productivity -- DONE

## **DONE**

**Stories are Closed within a Sprint**

**Feature Complete and Tested**

**Code Complete and Clean**

**No Known Defects**

**Production Ready**

**Approved by the Product Owner “Done-Done”**



Sutherland, Cohn

# SPRINT REVIEWS

Always demo features whether they are complete or not

... and promise to fix the bugs

(DONE)

Let the team feel safe

Have only the ScrumMaster or tech lead do the demos

Anyone can demo, rotate



# PROJECT LEADERSHIP



Buy the best ALM tool as soon as you have budget for it  
First establish the Process – stay manual as long as possible

Publish all project information electronically

Use paper scrum boards as much as possible, as long as possible

-- Information radiators

-- Tactile and social interaction

Rotate the ScrumMaster role among team members

Encourage “emergent leadership”

# STAKEHOLDER MANAGEMENT

Assume the Agile sponsor still has the influence he/she had when you came in

Stay alert to power shifts, nurture relationships



Trust that C-level management trusts what you're doing

Feed them information on velocity, defects, Nokia test trends, impediments, sprint goals

Let them know progress along the roadmap

- release burndown/up
- capabilities implemented this sprint, release



Update RTM date estimates after every sprint

No more than one “shock” per release (Cohn)

Assume adequate server capacity, speed, availability

- Sharing servers with other projects => false positives: latency, crashes

Get your own dev, CM, integration, build, QA environment

Daily builds and continuous integration are very difficult for distributed teams

...so once a week will be OK

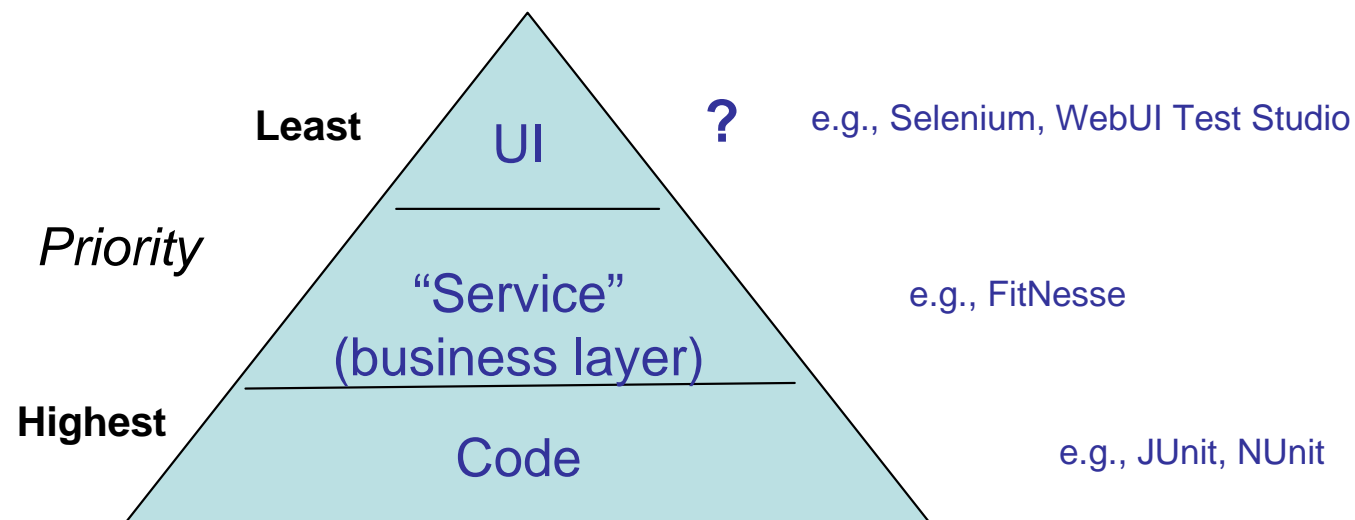
SCM: Bazaar, ClearCase, etc. ABM: Cruise Control, Anthill Pro, Hudson

Automated testing tools are really difficult to set up

...so we'll wait a while

Hyper-productivity isn't possible without automated testing

## Mike Cohn's Strategy for Automated Testing



# ORGANIZATION

“Urgent Need: Sr. QA with experience in an Agile Scrum environment”

[Apply for this job](#)

- “QA”s often are just manual UI / UE testers
- Manual testers don’t easily transition to automated testing

Urgent Need: Test Engineers

- hands-on experience installing and using automated testing tools
- e.g., can write FitNesse fixtures

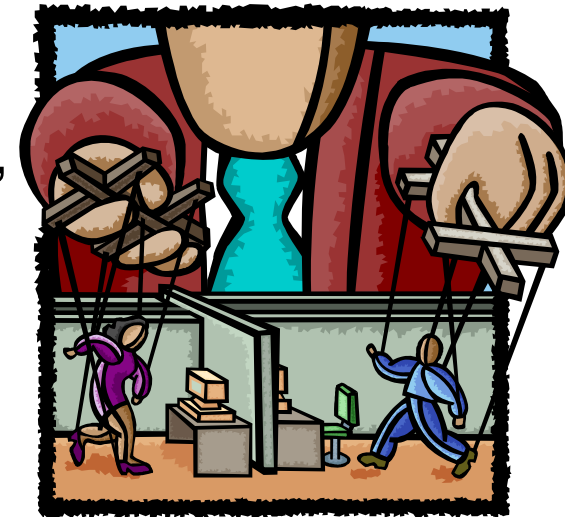
A PM gets assigned to work with you

Give the PM useful work to do

infrastructure impediments, licenses, ALM tool, stats

# CULTURE

- “We need regular status meetings and reports”
- “My CEO wants to see an end-to-end project plan”
- “We need complete requirements”
- “You haven’t defined the end state”
- “We need a resource-loaded project plan”
- “We need to see actual hours by task and developer”
- “When do you assign tasks?”



Keep information flowing:  
roadmap status, capabilities  
velocity, burndown,  
(EVA, Gantt)

## Fear at the Edge of Chaos

### Relapse Modes

- Waterfall Thinking
- Command and Control
- Denial of Physical Laws

Schwaber

# LEARNING

Trust that kaizen will happen naturally

After every sprint, make a list of what went wrong

Do a mindful retrospective after every sprint

Do a deep-dive retrospective after every release

