

Kanban For Software Engineering

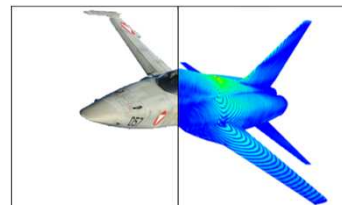
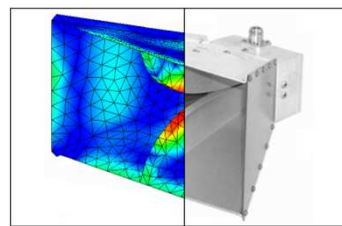
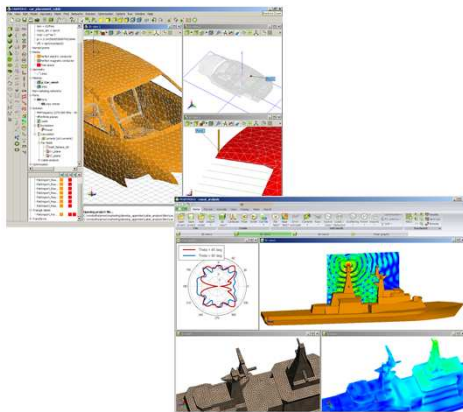


Jaco van der Merwe
Electromagnetic Software & Systems
(EMSS)

18/8/2010

jvdmerwe@emss.co.za

FEKO



www.feko.info



General Applications of FEKO

Antennas

Microwave components

Antenna placement

Antenna coupling

Scattering analysis

www.feko.info

FEKO
Comprehensive Electromagnetic Solutions

Detailed description: This slide illustrates various applications of FEKO software. It features several 3D models: a parabolic antenna with a color-coded radiation pattern, a microwave component with a cross-section showing internal fields, an aircraft with an antenna mounted on its fuselage, a close-up of an antenna on a fuselage section, a helicopter with a color-coded scattering analysis, and a large aircraft with a full-body scattering analysis. The FEKO logo and website are at the bottom.

General Applications of FEKO

Cable coupling analysis

Radiation hazard analysis

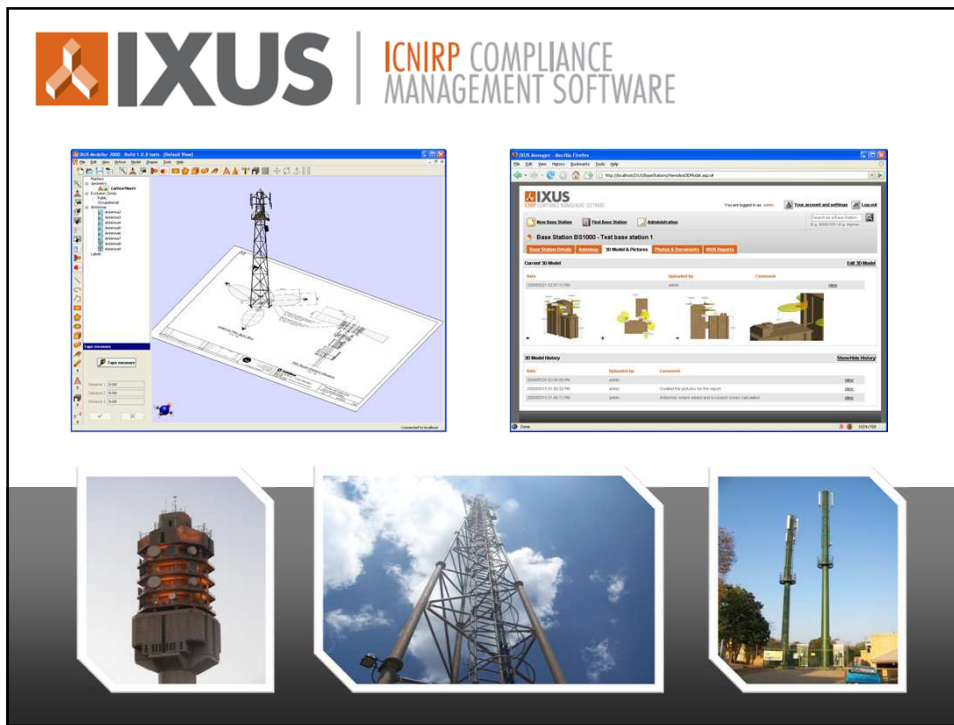
Bio-electromagnetic interaction

Mast configuration

www.feko.info

FEKO
Comprehensive Electromagnetic Solutions

Detailed description: This slide shows more FEKO applications. It includes: a car chassis for cable coupling analysis, a radiation hazard analysis showing a helicopter on a terrain with a color-coded field distribution and a legend, a human head model for bio-electromagnetic interaction, a lattice tower structure, and a ship's mast configuration. The FEKO logo and website are at the bottom.



Agenda

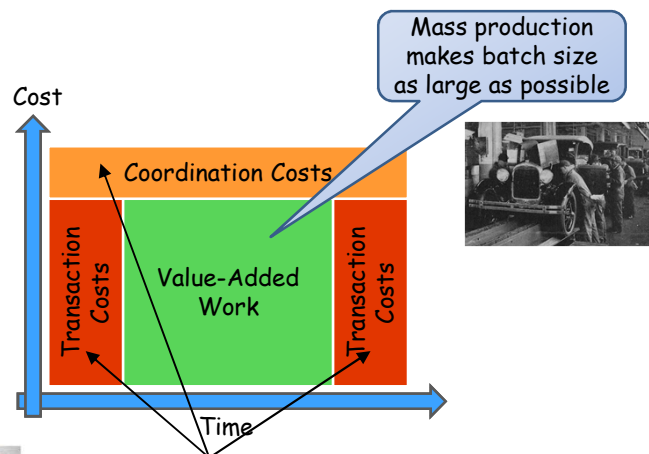
- Introduction
- The Kanban System
- Setting Up A Kanban Board (Card Wall)
- Kanban In A Nutshell
- Resources
- Acknowledgments

18/8/2010

Jaco van der Merwe

7

An Economic Model of Work

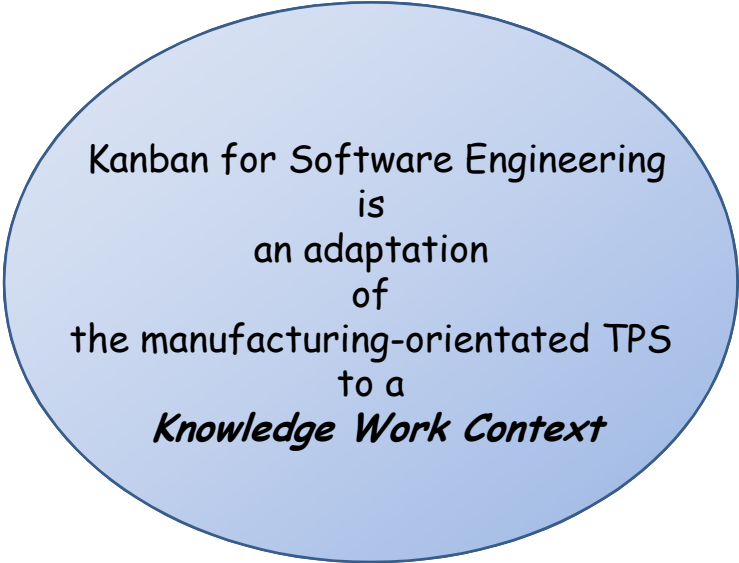


Taiichi Ohno

18/8/2010

Jaco van der Merwe

8



Kanban for Software Engineering
is
an adaptation
of
the manufacturing-orientated TPS
to a
Knowledge Work Context

18/8/2010 Jaco van der Merwe 9

Development Processes

- What is a process?
- Why do you need a process?
- What happens when you have too little process?
 - Unpredictability
 - Repeated errors
 - Wasted effort
- What happens when you have too much process?
 - Reduced performance
 - Increased costs
 - Stifled innovation and creativity
- Is there a "one size fits all" process?

18/8/2010 Jaco van der Merwe 10

Development Processes

- Use a method that would let a new process evolve
- Kanban is an evolutionary change method
 - Installs incremental process improvement through repeated discovery and fixing of issues affecting process performance
 - Provides a systemic way to achieve sustainable pace
 - Utilizes a kanban pull system (small k), visualization and other tools described in this presentation

18/8/2010

Jaco van der Merwe

11

Prescriptive vs. Adaptive

- Kanban...
 - is *not* a software development lifecycle methodology
 - is *not* an approach to project management
 - requires that some process is already being followed
 - is applied to incrementally improve the underlying process
- Kanban provides freedom...
 - To create a tailored process optimized to a specific context
 - For people to think for themselves and be different
- Kanban provides the *tools* to explain and justify why being different is better and why it is the right choice in that context

18/8/2010

Jaco van der Merwe

12

Crowd Control For A Maze



No more than 5 people allowed inside the maze at any time



kan-ban (signal cards)

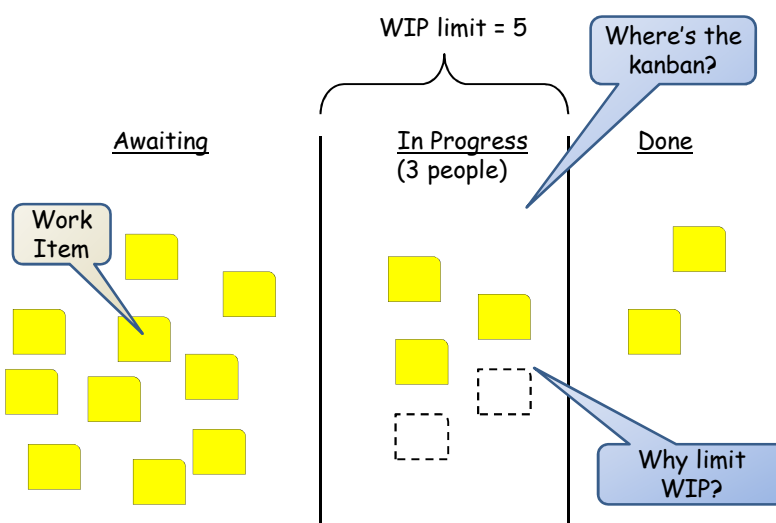


18/8/2010

Jaco van der Merwe

15

How To limit Work-In-Progress (WIP)?



18/8/2010

Jaco van der Merwe

16

The Rules of Kanban

- A number of kanban equivalent to the (agreed) capacity of a system are placed in circulation
- One card attaches to one work item
- Each card acts as a signaling mechanism
- A new work item can only be started when a free card is available
- A free card is attached to a work item and follows it as it flows through the system
- When there are no more free cards, no new work can be started
- Any work waiting to start must queue until a card becomes available
- When a work item is completed, its card is detached and recycled
- With a card now free a new piece of work queuing can be started

18/8/2010

Jaco van der Merwe

17

Kanban is a Pull System

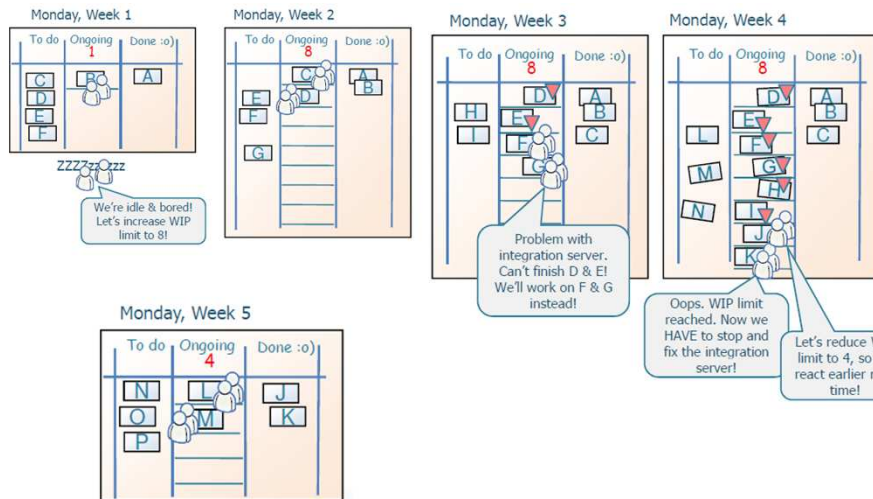
- New work pulled into the system only when there is capacity to handle it
- No work is pushed into the system based on demand
- A pull system cannot be overloaded
- Proviso: the capacity as determined by the number of kanban in circulation has been set appropriately

18/8/2010

Jaco van der Merwe

18

Experimenting With WIP Limits

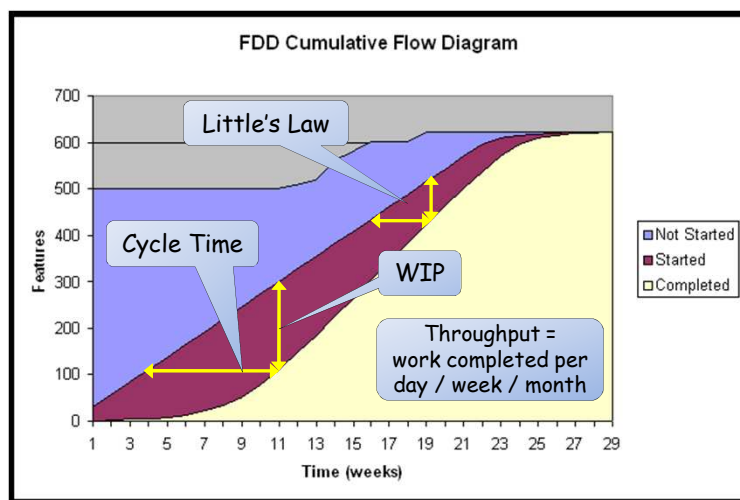


18/8/2010

Jaco van der Merwe

19

Cumulative Flow Diagram



18/8/2010

Jaco van der Merwe

20

Throughput, WIP & Cycle Time

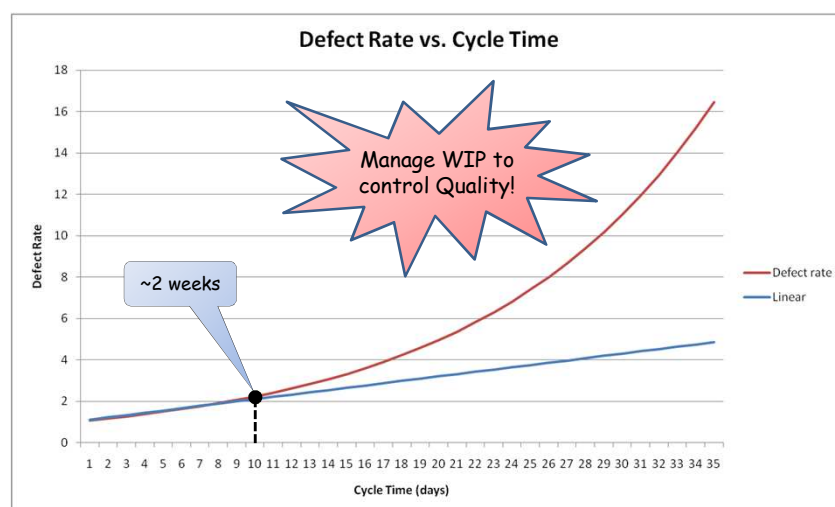
- Cycle time = time from work started to completed (flow time)
- Lead time = time from customer request to work delivered
- Little's Law: $\text{Cycle Time} = \text{WIP} / \text{Throughput}$ (in stable system)
 - For a given throughput (with everybody busy at their jobs), an increase in WIP means an increase in cycle time
 - Increase WIP today = increase in the time to deliver that work in the future
 - Caveat: reducing WIP can also reduce throughput
- Manage Cycle Time:
 - Control WIP (by using WIP limits)
 - Increase Throughput
 - Which is the easier one to control?

18/8/2010

Jaco van der Merwe

21

Cycle Time and Quality



18/8/2010

Jaco van der Merwe

22

Shorter Cycle Times

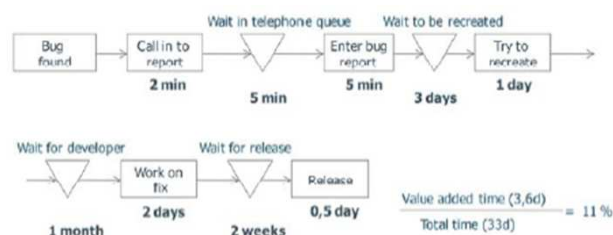
- Short cycle times enable projects to:
 - Deliver work early
 - Deliver work regularly
 - Obtain early and quick feedback
 - Have lower defect rates (higher quality)
 - Increase visibility of progress
- Do short cycle times necessarily lead to short lead times?
 - New version of product released/deployed once a year?
 - Average lead time = 1.5 years
 - Cycle time could be short internally though
 - New version of product released/deployed every 2 months
 - Average lead time = 3 months
 - Forces a short cycle time

18/8/2010

Jaco van der Merwe

23

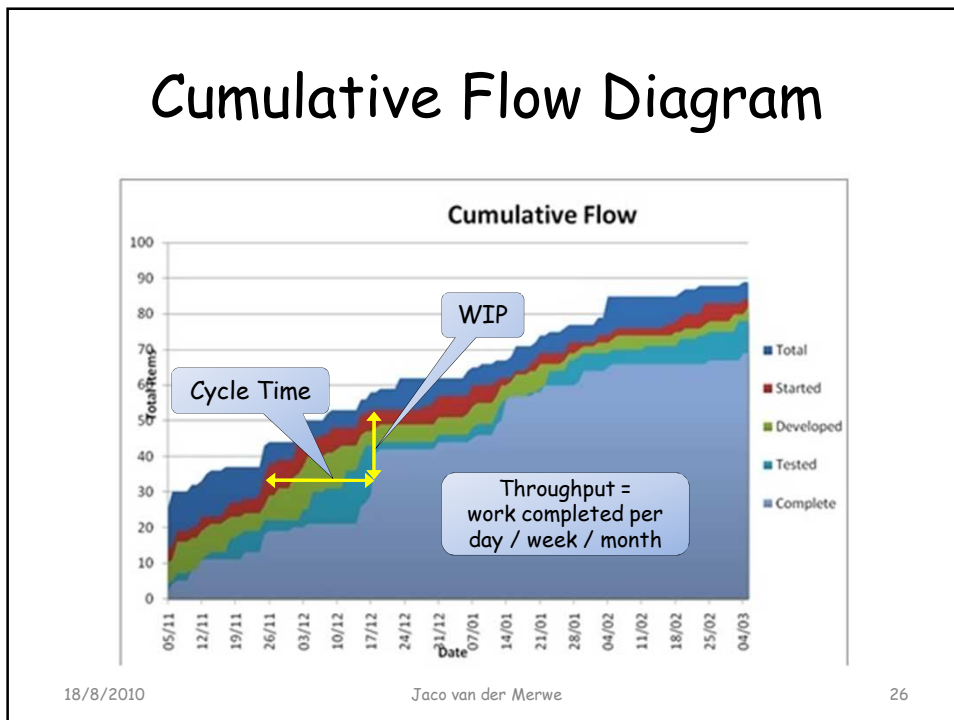
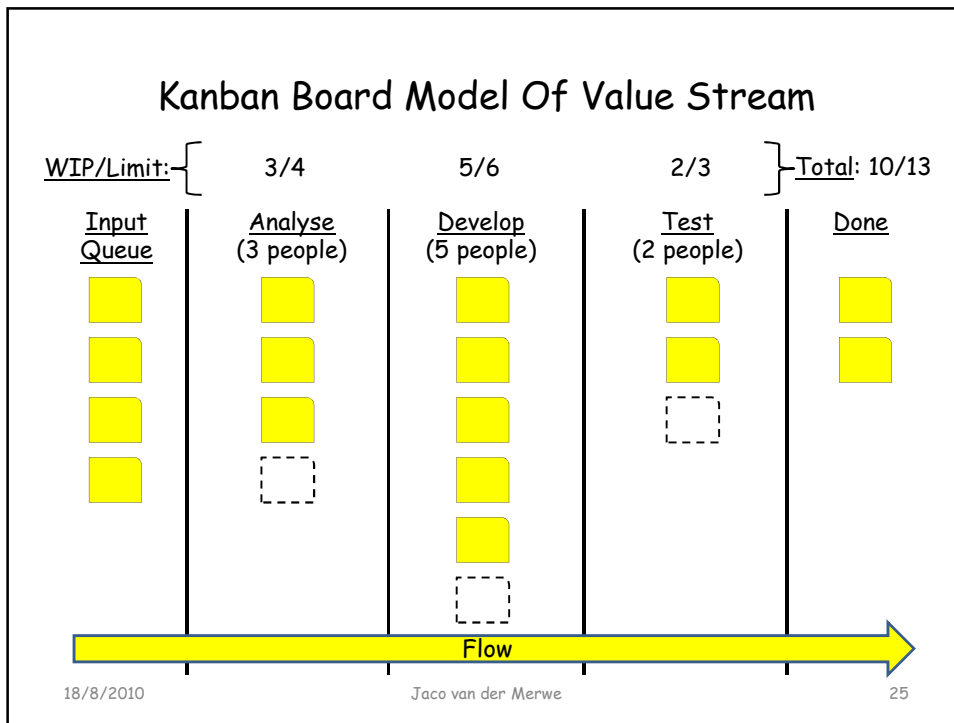
Value Streams, Maps & Flow

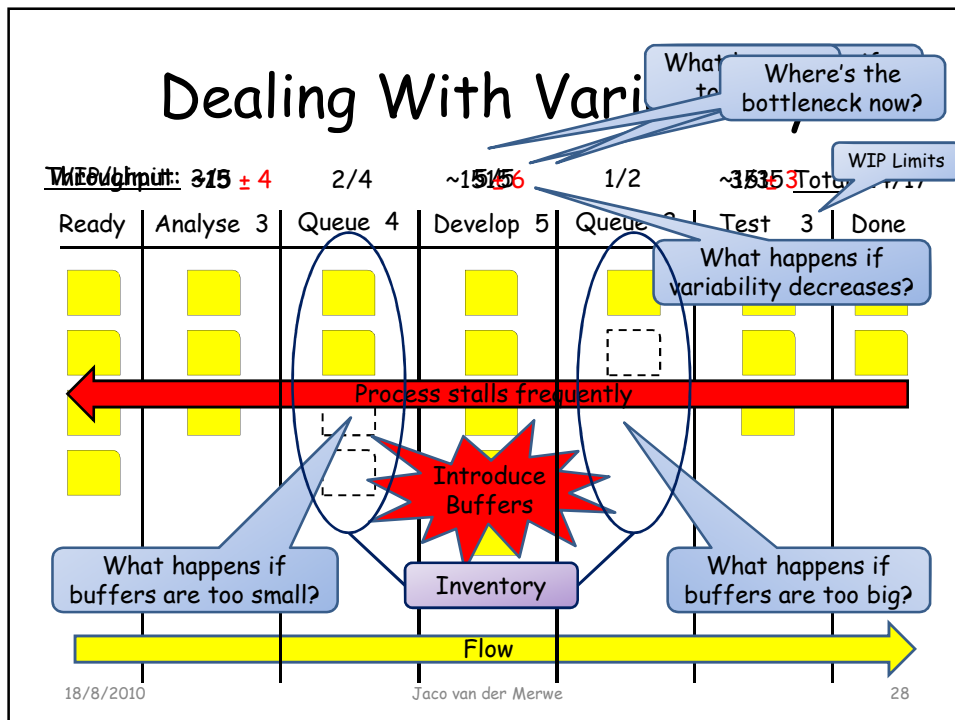
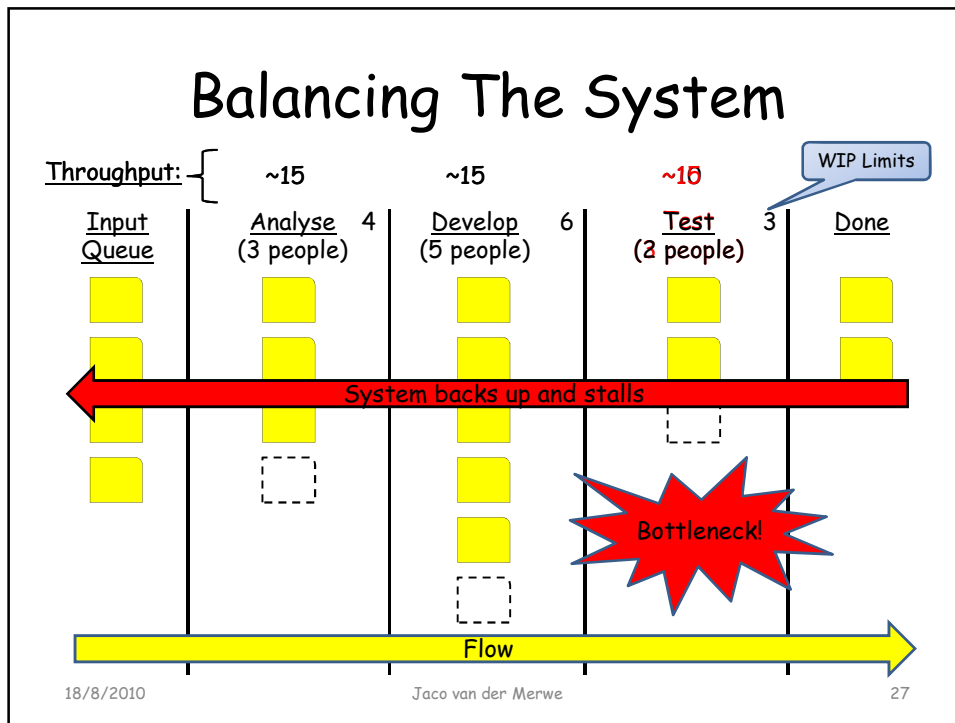


18/8/2010

Jaco van der Merwe

24





Dealing With Variability

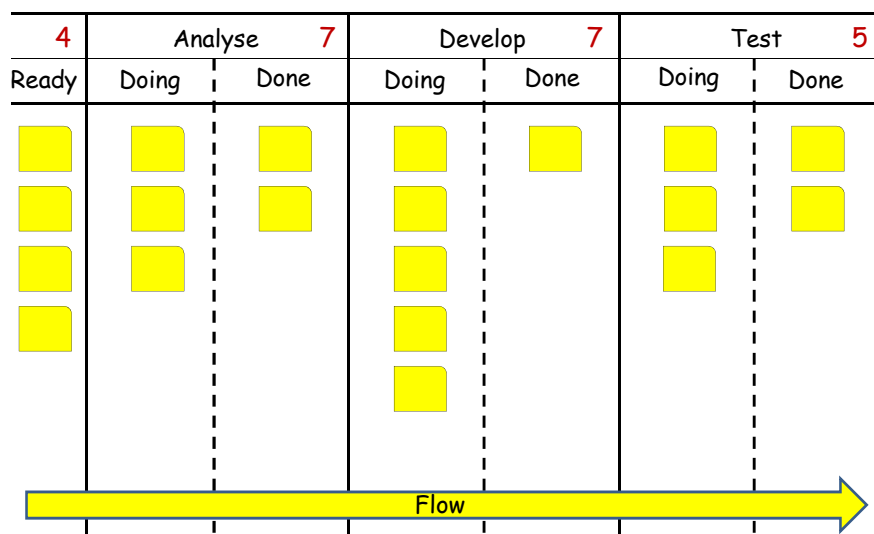
- Buffer states are non-value-adding processing time
- Queues are there for the purpose of smooth flow
 - **Too big:** Smooth flow but increases Inventory, WIP and Cycle Time
 - **Too small:** Stalled flow which increases Cycle Time
- Under normal conditions of smooth flow queues should be operating below their limits
- Queue filling up is a leading indicator of stalling
- Reduce variability => smaller queues => shorter Cycle Times

18/8/2010

Jaco van der Merwe

29

Shared Completion Queues

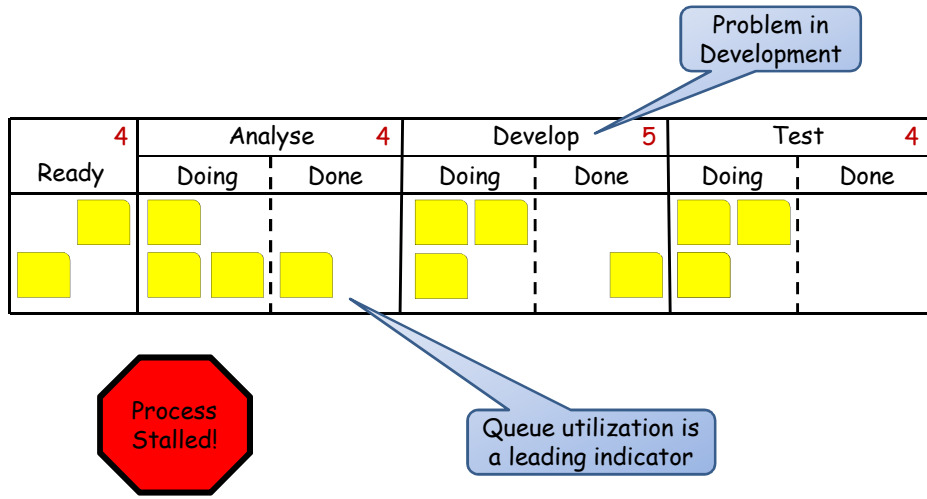


18/8/2010

Jaco van der Merwe

30

Completion Queue Stalling

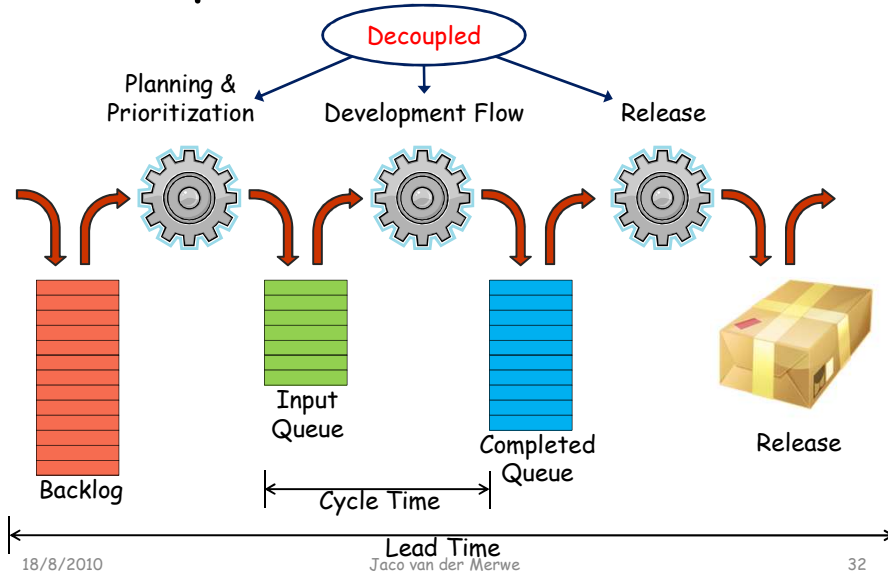


18/8/2010

Jaco van der Merwe

31

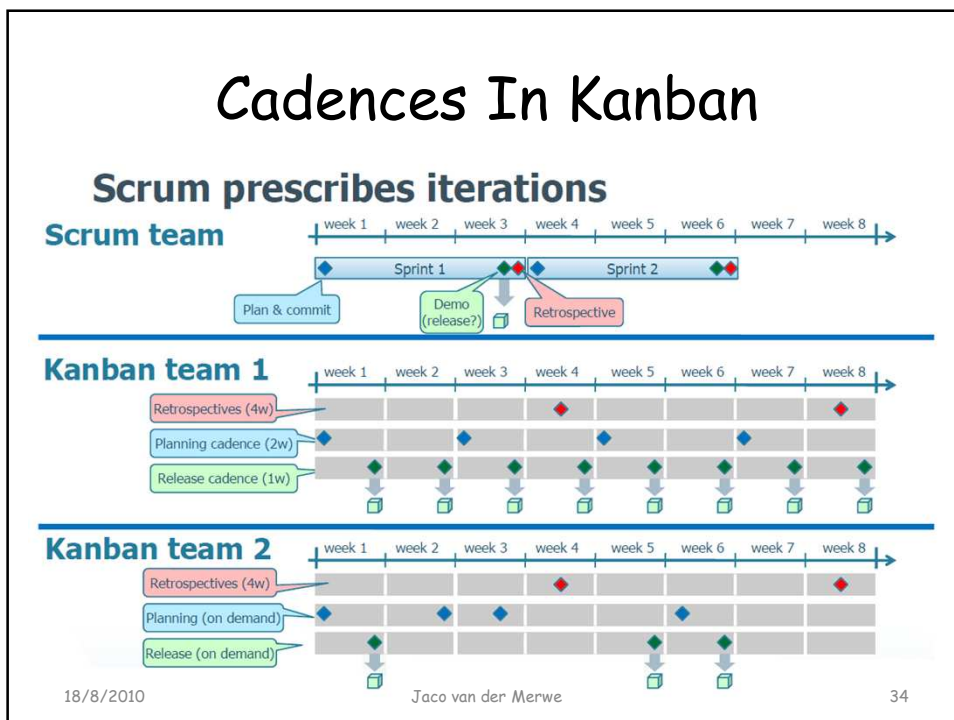
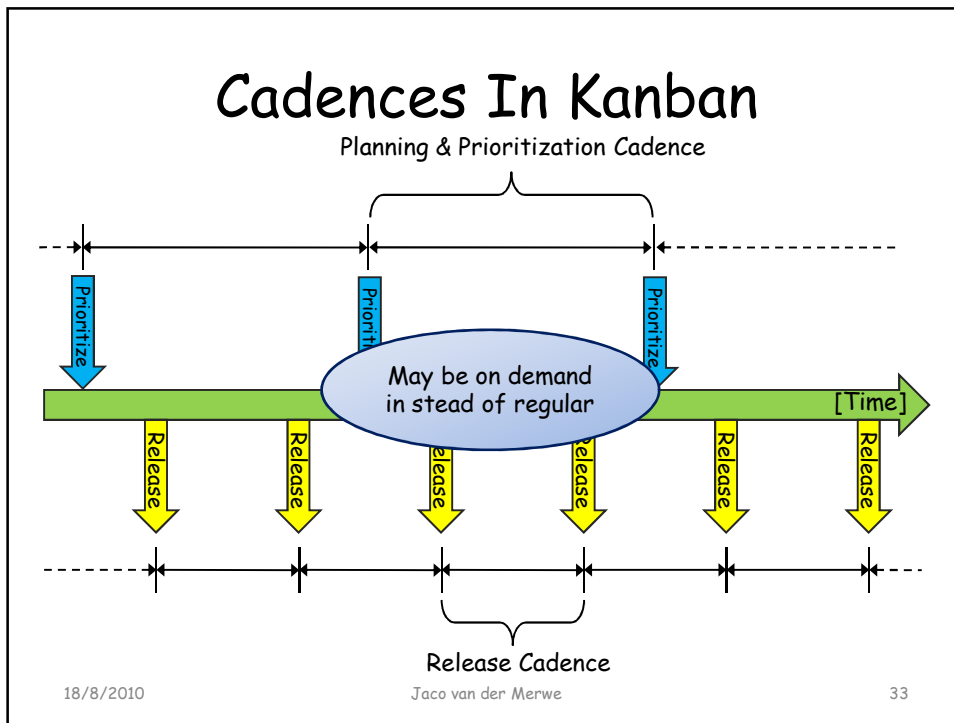
Decoupled Activities in Kanban



18/8/2010

Jaco van der Merwe

32



What Have We Learned So Far?

- Limiting WIP with kanban
- How a pull system prevents overloading
- Cumulative Flow Diagrams
- WIP, Throughput, Cycle Time, Lead Time
- Relationship between WIP, Throughput & Cycle Time
- Relationship between Cycle Time and Quality
- Benefits of shorter Cycle Times
- Value streams and flow
- Identifying bottlenecks and balancing throughput
- Managing variability by means of buffer states
- Decoupled activities and cadences in Kanban

18/8/2010

Jaco van der Merwe

35

SETTING UP A KANBAN BOARD



18/8/2010

Jaco van der Merwe

36

Map The Value Stream

- Change as little as possible
- Resist temptation to change workflow, job titles, roles, responsibilities, specific working practices
- Show the activities that happen to the work
- Draw columns on board to represent the activities performed in the order they are performed
 - Typically flows from left to right
 - Some teams use flow from bottom to top
 - Initially draw columns/rows with erasable markers
 - Changes will typically be made during first few weeks
 - Once workflow design has stabilized use thin precision (3mm) whiteboard tape

18/8/2010

Jaco van der Merwe

37

Identify Work Item Types

- Each card represents a discrete unit of customer-valued work
- Examples of Work Item Types:
 - Requirement, Feature, User Story
 - Use Case, Change Request, Production Defect
 - Maintenance, Refactoring, Bug
 - Improvement, Blocking Issue
- May be hierarchical
 - Epic, high level feature, related collection of user stories
- Steps may be different for different work item types
- Decide how to visually communicate work item types:
 - Use colour, shape, symbols, annotations, swim lanes, ...

18/8/2010

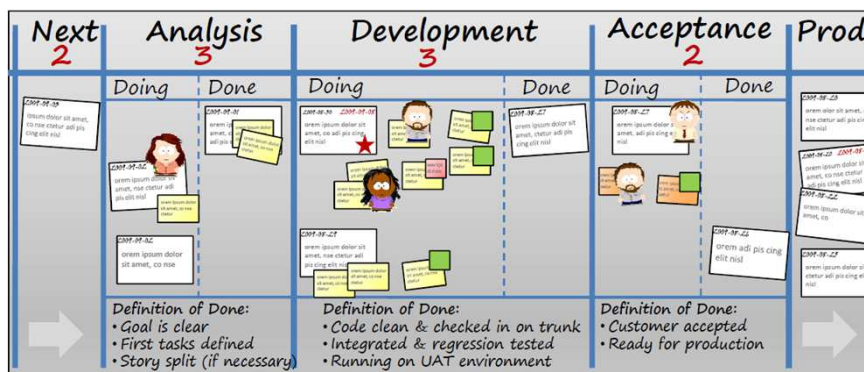
Jaco van der Merwe

38

Set WIP Limits & Add Buffers

- **Approach A**
 - Do not try to second guess the locations of bottlenecks or sources of variability that will require buffers
 - Rather implement the system and wait for the bottleneck to reveal itself and then introduce a buffer
 - Variant: initially set WIP limits fairly loosely so that variability, waste and bottlenecks do not have a significant impact on the pull system when it is first implemented
- **Approach B**
 - Each stage should be buffered
 - The activity steps should have tight WIP limits
 - Bottlenecks and variability will reveal themselves by how full the buffers become
 - Small changes can then be made to reduce buffer sizes and eventually eliminate unnecessary buffers
- Currently not enough evidence available to suggest which approach is better

Board with WIP Limits & Buffers

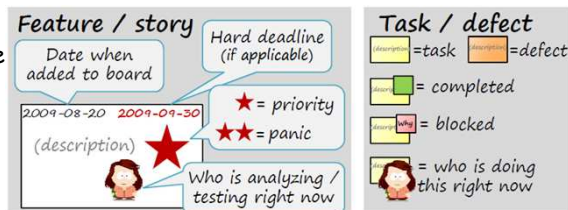


Anatomy Of A Card

- Information on each card is important
 - Facilitates the pull system
 - Empowers individuals to make their own pull decisions
- Information may vary by work item type or class of service

- Make use of:

- Card colour & shape
- Symbols or icons
- Text on card
- Stickers
- Swim lanes



- Some information remains the same as card flows (static)
- Some information changes with workflow (dynamic)

18/8/2010

Jaco van der Merwe

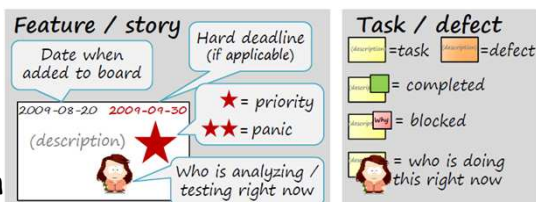
41

Anatomy Of A Card

- Static information
 - Electronic tracking number
 - Title or description
 - Work item type
 - Class of service
 - Entry date
 - Delivery date

- Dynamic information

- Current assignee
- Days in column/activity
- Item is overdue
- Item is impeded
- Returned due to defect



18/8/2010

Jaco van der Merwe

42

Define A Pull Policy

- Defines which work items to pull first when capacity becomes available
- Examples:
 - Take any item
 - Always take the top item
 - Always take the oldest item
 - 20% on maintenance items, 80% on new features
 - Split capacity evenly between product A and product B
 - Always take red items first
- Classes of service another useful approach

What to pull first

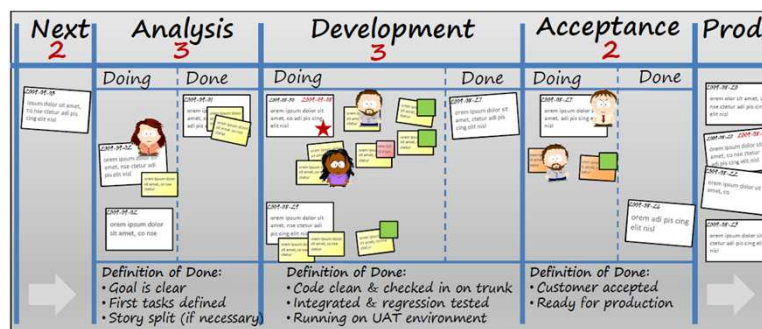
1. Panic features ★★
(should be swarmed and kept moving. Interrupt other work and break WIP limits as necessary)
2. Priority features ★
3. Hard deadline features
(only if deadline is at risk)
4. Oldest features

18/8/2010

Jaco van der Merwe

43

Example: Complete Card Wall



Feature / story

Date when added to board: 2009-08-20, 2009-09-30

Hard deadline (if applicable): ★ = priority, ★★ = panic

Who is analyzing / testing right now: [Icon]

Task / defect

[Yellow] = task, [Orange] = defect, [Green] = completed, [Red] = blocked, [Icon] = who is doing this right now

What to pull first

1. Panic features ★★
(should be swarmed and kept moving. Interrupt other work and break WIP limits as necessary)
2. Priority features ★
3. Hard deadline features
(only if deadline is at risk)
4. Oldest features

18/8/2010

Jaco van der Merwe

44

Electronic Card Walls

- Essential for distributed teams or when members sometimes work from home
- A number of products emerging:
 - Lean Kit Kanban; Agile Zen; Target Process
 - Silver Catalyst; RadTrack; Kanbanery
 - VersionOne; Greenhopper for Jira; Flow.io
- Necessary for teams that aspire to higher levels of organizational maturity
 - Quantitative management
 - Organizational process performance (comparing the performance across kanban systems, teams or projects)
 - Causal analysis and resolution (root cause analysis based on statistically sound data)

18/8/2010

Jaco van der Merwe

45

KANBAN IN A NUTSHELL

18/8/2010

Jaco van der Merwe

46

Kanban In A Nutshell

- **Visualize the Workflow**
 - Represent the work items and the workflow on a card wall or electronic board.
- **Limit Work In Progress (WIP)**
 - Set agreed upon limits to how many items may be in progress at each workflow state
- **Measure and Manage Flow**
 - Track work items to see if they are proceeding at a steady, even pace.
 - Make cycle time as small and predictable as possible
- **Make Process Policies Explicit**
 - Agree upon and post policies about how work will be handled
- **Use Models to Evaluate Improvement Opportunities**
 - Adapt the process using ideas from Systems Thinking
 - Gathering data on system performance: flow time, WIP, delivery throughput, etc., provides scientific insight into opportunities for improvements

18/8/2010

Jaco van der Merwe

47

Benefits

- **Optimization of Existing Processes**
 - Introduction of visualization and the limiting of work-in-progress (WIP) will catalyze change with minimal disruption
- **Delivering with Higher Quality**
 - Limiting work-in-progress and defining policies for work prioritization will bring greater focus on quality
 - Policies can also address quality criteria directly
- **Improved Cycle Time Predictability**
 - Correlation between WIP, cycle time and defect rates
 - Limiting WIP makes cycle times dependable, keeps defect rates low
- **Improved Employee Satisfaction**
 - Kanban reduces context switching and pulls work at the rate the team can complete it
 - Working at a more even, predictable pace, means employees are never overloaded

18/8/2010

Jaco van der Merwe

48

Benefits

- **Providing Slack to Enable Improvement**
 - Creating slack in the value chain improves responsiveness to urgent requests and bandwidth to enable process improvement and quality improvement
- **Simplified Prioritization**
 - Kanban enables fast re-prioritization to accommodate changes in the market
- **Transparency on the System and its Operation**
 - Improved visibility builds trust with customers and managers
 - Shows the effects of actions or inactions
 - Results in improved collaboration
- **Enables Emergence of a "High-Maturity" Organization**
 - As improvements are implemented, organizational maturity improves leading to better decision making and improved risk management
 - Risk, managed appropriately, brings predictable results

18/8/2010

Jaco van der Merwe

49

Resources

- **People**
 - David Anderson, Henrik Kniberg, Corey Ladas, Jeff Patton, Mattias Skarin
- **Books/Articles**
 - *"Kanban: Successful Evolutionary Change for Your Technology Business"*, David Anderson
 - *"Getting Started with Kanban for Software Development"* DZone Refcard, David Anderson & Janice Linden-Reed (free download on DZone)
 - *"Kanban and Scrum - Making The Most of Both"*, Henrik Kniberg & Mattias Skarin (free download on InfoQ)
 - *"Converting a Scrum Team to Kanban"*, Mattias Skarin (available from his blog)
- **Blogs**
 - David J. Anderson & Associates Blog
 - Henrik Kniberg's Blog
 - Mattias Skarin's Blog
 - Jeff Patton's Holistic Product Design & Development
 - Lean Software Engineering Blog (Corey Ladas)

18/8/2010

Jaco van der Merwe

50

Acknowledgments

- Card wall example
 - From "*Kanban Kickstart Example*",
Henrik Kniberg

18/8/2010

Jaco van der Merwe

51