INF5180: Software Product- and Process Improvement in Systems Development

Part 02:

Processes and Process Modeling

(Section B)

Spring 2010

UNIVERSITETET I OSLO

Dr. Dietmar Pfahl

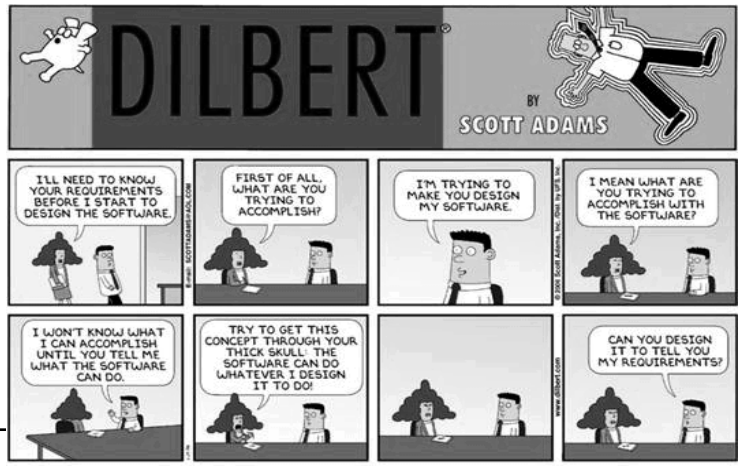
email: dietmarp@simula.no

”Light-Weight” Processes

(Evolutionary Development)

Copyright 2010 © Dietmar Pfahl

Requirements and Customers



© Scott Adams, Inc./Dist. by UFS, Inc.

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.

http://www.agilemanifesto.org/

Copyright 2010 © Dietmar Pfahl

Extreme Programming

- Origin: Kent Beck, Ward Cunningham, Ron Jeffries (end of 1990s)
- Idea: “light weight” process model, agile process
- Characteristic:
  - “Minimum” of accompanying measures (documentation, modeling , …)
  - Team orientation (e.g., common responsibility for all development artifacts)
  - Small teams (12-14 persons)
  - Involvement of user/client at an early stage
  - Social orientation
- Scope: Prototype projects, small projects, low criticality of the results

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

XP – Rules and Practices

http://www.extremeprogramming.org/rules.html

Planning

User stories are written (by the customer!).
Release planning creates the schedule.
Make frequent small releases.
The Project Velocity is measured.
The project is divided into iterations.
Iteration planning starts each iteration.
Move people around.
A stand-up meeting starts each day.
Fix XP when it breaks.

Designing

Simplicity.
Choose a system metaphor.
Use CRC* cards for design sessions.
Create spike solutions to reduce risk.
No functionality is added early.
Refactor whenever and wherever possible.

Coding

The customer is always available.
Code must be written to agreed standards.
Code the unit test first.
All production code is pair programmed.
Only one pair integrates code at a time.
Integrate often.
Use collective code ownership.
Leave optimization till last.
No overtime.

Testing

All code must have unit tests.
All code must pass all unit tests before it can be released.
When a bug is found (acceptance) tests are created.
Acceptance tests are run often and the score is published.

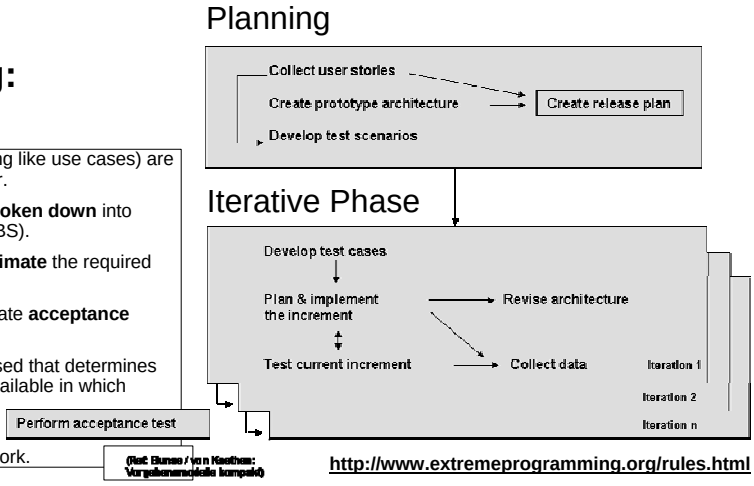
Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

* CRC = Class Responsibility Collaborator

Extreme Programming: Overview

1. User **stories** (something like use cases) are written by the customer.
2. Complex stories are **broken down** into simpler ones (like a WBS).
3. Stories are used to **estimate** the required amount of work.
4. Stories are used to create **acceptance tests**.
5. A **release plan** is devised that determines which stories will be available in which release.
6. Don't hesitate to change what doesn't work.



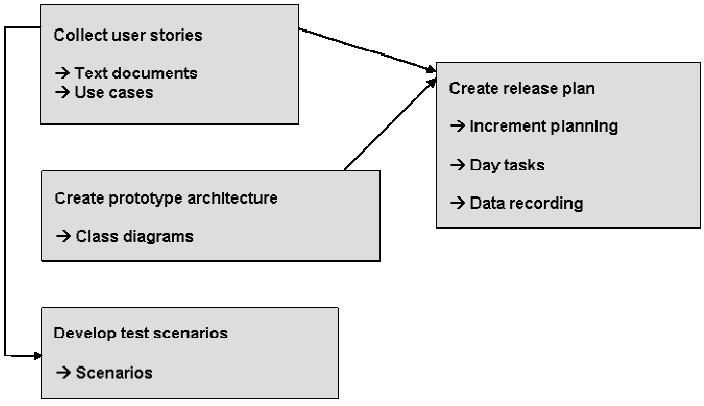
http://www.extremeprogramming.org/rules.html

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

Extreme Programming: Planning

1. Each release is preceded by a **release planning meeting**.
2. Each day begins with a **stand-up meeting** to share problems and concerns.
3. CRC cards are used for **design.** [XP and CRC were created by the same person, Kent Beck.]
4. **Spike solutions** are done to assess risks.
5. The **customer** is always available.



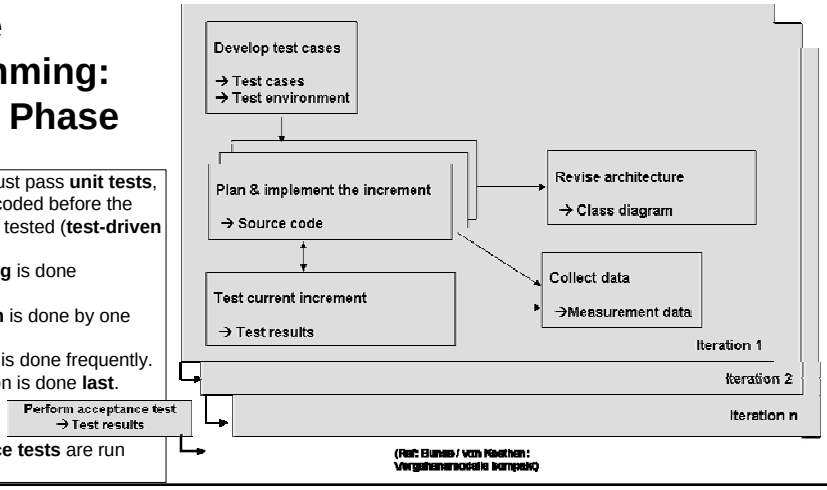
(Ref: Bunse / von Knethen: Vorgehensmodelle kompakt)

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

Extreme Programming: Iterative Phase

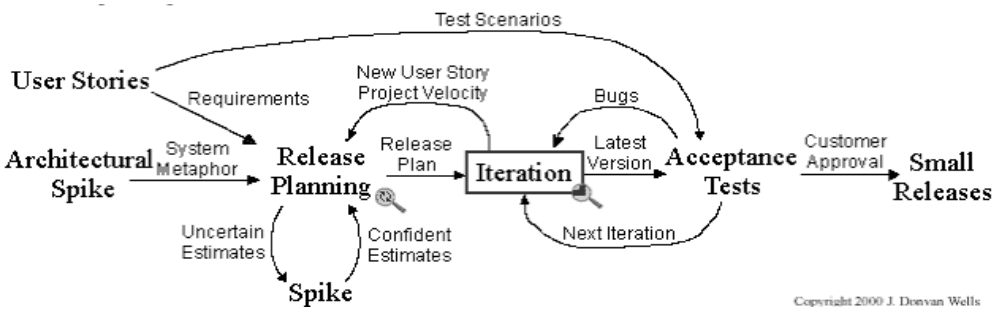
1. All code must pass **unit tests**, which are coded before the code being tested (**test-driven design**).
2. **Refactoring** is done constantly.
3. **Integration** is done by one pair.
4. Integration is done frequently.
5. Optimization is done **last**.
6. **Acceptance tests** are run often.



Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

Extreme Programming – Evolutionary Process Model



Copyright 2010 © Dietmar Pfahl

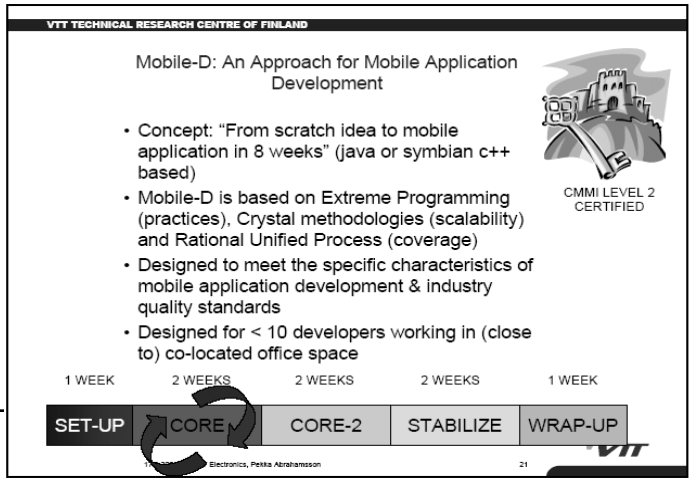
UNIVERSITETET I OSLO

Mobile-D

Defined by VTT for the mobile phone industry in Finland

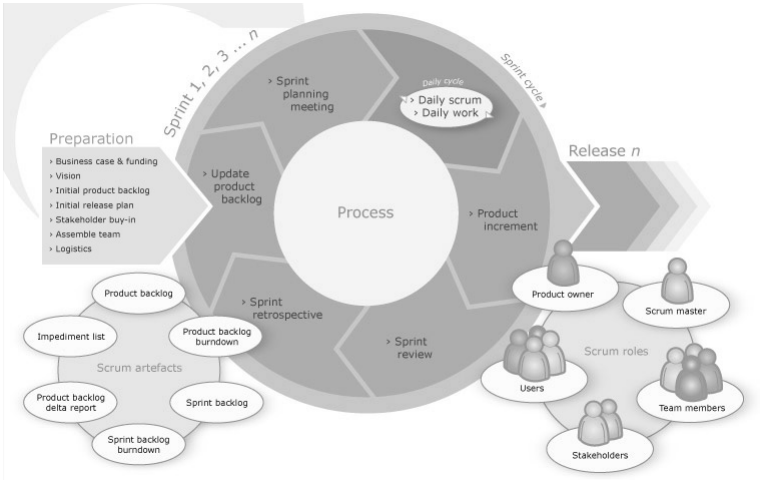
More on Crystal methodologies can be found at:

http://alistair.cockburn.us/index.php/Crystal_methodologies



UNIVERSITETET I OSLO

Scrum



Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

http://www.scrumforteamsystem.com/processguidance/v1/Scrum/Scrum.html

Scrum – Roles: ”Pigs” and ”Chicken”

"Pig" roles

- Pigs are the ones committed to the project in the Scrum process; they are the ones with "their bacon on the line".
  - Product Owner
  - Scrum Master (or Facilitator)
  - Team

"Chicken" roles

- Chicken roles are not part of the actual Scrum process, but must be taken into account.
  - Users
  - Stakeholders (customers, vendors)
  - Managers
- Note: An important aspect of an Agile approach is the practice of involving users, business and stakeholders into part of the process. It is important for these people to be engaged and provide feedback into the outputs for review and planning of each sprint.

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

Scrum – Roles

"Pig" roles:

- Product Owner
  - The Product Owner represents the voice of the customer ensuring that the Team works on the right things from a business perspective.
  - The Product Owner writes user stories, prioritizes them, then places them in the product backlog.
- Scrum Master (or Facilitator)
  - Scrum is facilitated by a ScrumMaster, whose primary job is to remove impediments to the ability of the team to deliver the sprint goal.
  - The ScrumMaster is not the leader of the team (as they are self-organizing) but acts as a buffer between the team and any distracting influences.
  - The ScrumMaster ensures that the Scrum process is used as intended. The ScrumMaster is the enforcer of rules.
- Team
  - The team has the responsibility to deliver the product.
  - A team is typically made up of 5–9 people with cross-functional skills to do the actual work (designer, developer, tester, etc.).

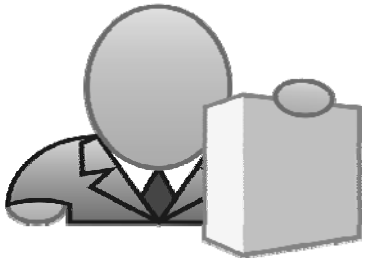
"Chicken" roles:

- Users
  - The software is being built for someone.
- Stakeholders (customers, vendors)
  - The people that will enable the project, and for whom the project will produce the agreed-upon benefit(s) which justify it. They are only directly involved in the process at sprint reviews.
- Managers
  - People that will set up the environment for the product development organizations.

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

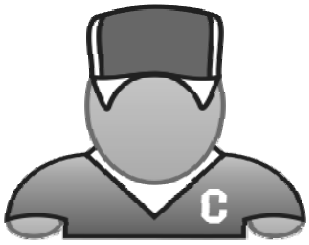
Product Owner



- Define the features of the product
- Decide on release date and content
- Be responsible for the profitability of the product (ROI)
- Prioritize features according to market value
- Adjust features and priority every iteration, as needed
- Accept or reject work results

Copyright 2010 © Dietmar Pfahl

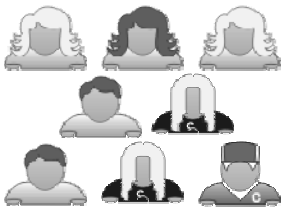
The ScrumMaster



- Represents management to the project
- Responsible for enacting Scrum values and practices
- Removes impediments
- Ensure that the team is fully functional and productive
- Enable close cooperation across all roles and functions
- Shield the team from external interferences

Copyright 2010 © Dietmar Pfahl

The Team

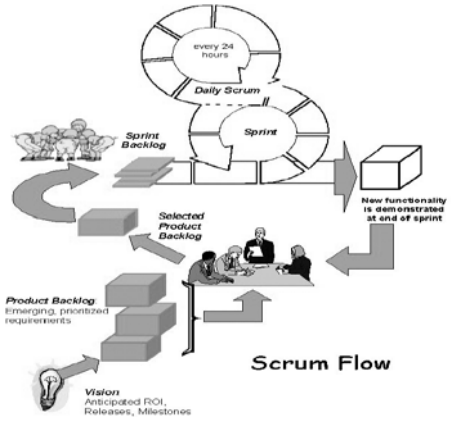


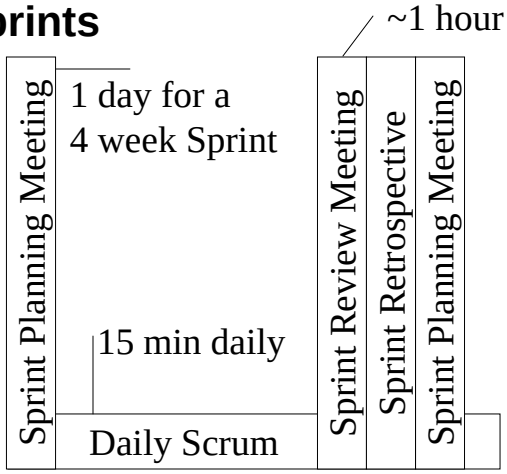
- Typically 5-9 people
- Cross-functional:
  - Programmers, testers, user experience designers, etc.
- Members should be full-time
  - May be exceptions (e.g., database administrator)
- Teams are self-organizing
  - Ideally, no titles but rarely a possibility
- Membership should change only between sprints

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

Scrum Iterations and Sprints





Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

Scrum – Meetings

- Daily Scrum
  - Each day during the sprint, a project status meeting occurs. This is called a "scrum", or "the daily standup". Daily scrum guidelines:
    - The meeting starts precisely on time. Often there are team-decided punishments for tardiness (e.g. money, push-ups, hanging a rubber chicken around your neck)
    - All are welcome, but only "pigs" may speak
    - The meeting is time-boxed (15 minutes) regardless of the team's size
    - All attendees should stand (it helps to keep meeting short)
    - The meeting should happen at the same location and same time every day
  - During the meeting, each team member answers three questions:
    - What have you done since yesterday?
    - What are you planning to do by today?
    - Do you have any problems preventing you from accomplishing your goal?
    - It is the task of the ScrumMaster to remind the team of these questions.
- Sprint Planning Meeting
  - Select what work is to be done
  - Prepare the Sprint Backlog that details the time it will take to do that work
  - 8 hour limit
- Sprint Review Meeting
  - Review the work that was completed and not completed
  - Present the completed work to the stakeholders (a.k.a. "the demo")
  - Incomplete work cannot be demonstrated
  - 4 hour time limit
- Sprint Retrospective
  - All team members reflect on the past sprint.
  - Make continuous process improvement.
  - Two main questions are asked in the sprint retrospective: What went well during the sprint? What could be improved in the next sprint?
  - 3 hour time limit

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

Sprint Planning Meeting

- Team selects items from the product backlog they can commit to completing
- Sprint backlog is created
  - Tasks are identified and each is estimated (1-16 hours)
  - Collaboratively, not done alone by the ScrumMaster
- High-level design is considered

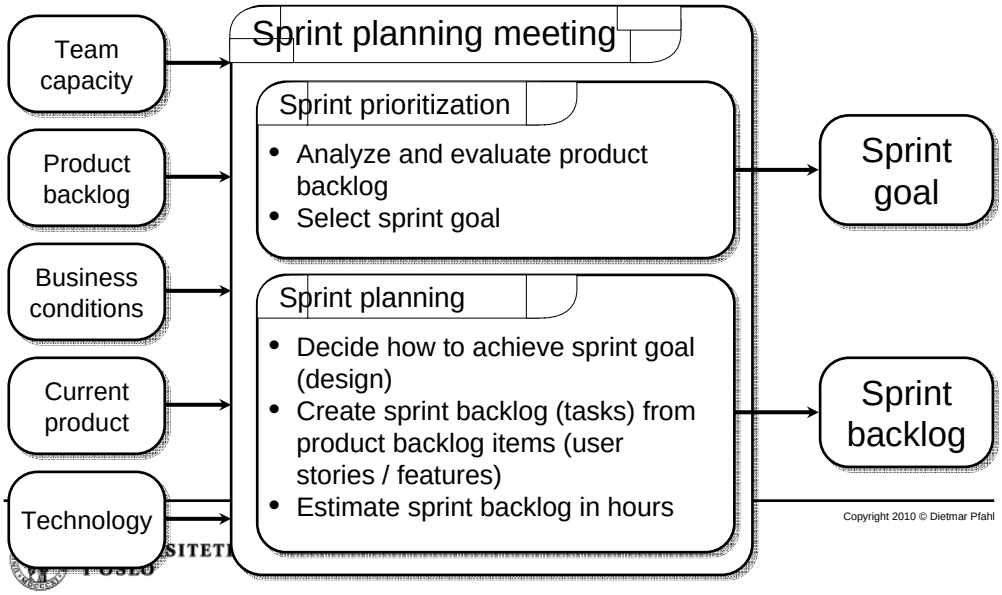
As a vacation planner, I want to see photos of the hotels.

→

Code the middle tier (8 hours)
Code the user interface (4)
Write test fixtures (4)
Code the foo class (6)
Update performance tests (4)

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO



Copyright 2010 © Dietmar Pfahl

Daily Scrum

- Parameters
  - Daily
  - 15-minutes
  - Stand-up

- Not for problem solving
  - Whole world is invited
  - Only team members, ScrumMaster, product owner, can talk
- Helps avoid other unnecessary meetings

Copyright 2010 © Dietmar Pfahl

Daily Scrum – 3 Questions

NB:

- These questions are not status reports for the ScrumMaster
- They are commitments in front of peers

1. What did you do yesterday?
2. What will you do today?
3. Is anything in your way?

Copyright 2010 © Dietmar Pfahl

Sprint Review Meeting

- Team presents what it accomplished during the sprint
- Typically takes the form of a demo of new features or underlying architecture
- Informal
  - 2-hour prep time rule
  - No slides
- Whole team participates
- Invite the world

Copyright 2010 © Dietmar Pfahl

Sprint Retrospective

- Periodically take a look at what is and is not working
- Typically 15–30 minutes
- Done after every sprint
- Whole team participates
  - ScrumMaster
  - Product owner
  - Team
  - Possibly customers and others

Copyright 2010 © Dietmar Pfahl

Scrum – Artifacts

Product backlog

- The product backlog is a high-level document for the entire project. It contains backlog items: broad descriptions of all required features, wish-list items, etc. It is the "What" that will be built. It is open and editable by anyone and contains rough estimates of both business value and development effort. Those estimates help the Product Owner to gauge the timeline and, to a limited extent, priority.
  - For example, if the "add spellcheck" and "add table support" features have the same business value, the one with the smallest development effort will probably have higher priority, because the return-on-investment is higher.
- The product backlog is property of the Product Owner. Business value is set by the Product Owner. Development effort is set by the Team.

Sprint backlog

- The sprint backlog is a greatly detailed document containing information about how the team is going to implement the requirements for the upcoming sprint. Tasks are broken down into hours, with no task being more than 16 hours. If a task is greater than 16 hours, it should be broken down further. Tasks on the sprint backlog are never assigned; rather, tasks are signed up for by the team members as they like.
- The sprint backlog is property of the Team. Estimations are set by the Team.

Burn down chart

- The burn down chart is a publicly displayed chart showing remaining work in the sprint backlog. Updated every day, it gives a simple view of the sprint progress.

Copyright 2010 © Dietmar Pfahl

Product Backlog

- The requirements
- A list of all desired work on the project
- Ideally expressed such that each item has value to the users or customers of the product
- Prioritized by the product owner
- Reprioritized at the start of each sprint

Copyright 2010 © Dietmar Pfahl

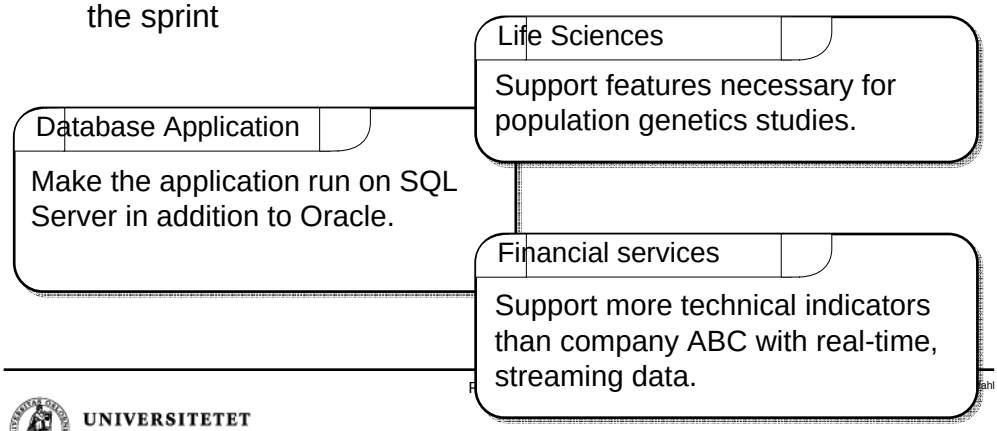
UNIVERSITETET I OSLO

Backlog item	Estimate
Allow a guest to make a reservation	3
As a guest, I want to cancel a reservation.	5
As a guest, I want to change the dates of a reservation.	3
As a hotel employee, I can run RevPAR reports (revenue-per-available-room)	8
Improve exception handling	8
...	30
...	50

UNIVERSITETET I OSLO

The Sprint Goal

- A short statement of what the work will be focused on during the sprint



Managing the Sprint Backlog

- Individuals sign up for work of their own choosing
  - Work is never assigned!
- Estimated work remaining is updated daily
- Any team member can add, delete or change the sprint backlog
- Work for the sprint emerges
- If work is unclear, define a sprint backlog item with a larger amount of time and break it down later
- Update work remaining as more becomes known

Copyright 2010 © Dietmar Pfahl

Sprint Backlog – Example

Tasks	Mon	Tues	Wed	Thur	Fri
Code the user interface	8	4	8		
Code the middle tier	16	12	10	4	
Test the middle tier	8	16	16	11	8
Write online help	12				
Write the foo class	8	8	8	8	8
Add error logging			8	4	

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

Sprint Backlog – Example (with more detail)

Core - Sprint 16 Backlog

Team: 8
Days: 10
Hours: 45
Max Hrs: 358

Sprint Start Date: 25.09.2006
Sprint End Date: 06.10.2006

Working Days Left: 10 9 8 7 6 5 4 3 2 1

Remaining Effort in Hours

#	Gemini	Item	Task	Status	P1	P2	25.09.2006	26.09.2006	27.09.2006	28.09.2006	29.09.2006	02.10.2006	03.10.2006	04.10.2006	05.10.2006	06.10.2006
							344	317	248	226	176	176	120	93	22	8
1		Complete XXX Page (continued)														
2		licensing (design only)		Complete	CJ	CC	40	37	32	16	16	16	8	0.5		
3		Field visibility (tech design)		In Progress	SB	CJ	20	20	20	20	20	20	20	20	4	1
4		Remove side panel & implement tabs		Complete	GB	AM	24	20								
5		Lookups (dropdowns, lookups, datepicker)														
6			Classification Code implementation	Complete	SB	GB	32	27	23	8						
7			Lookup control build	Complete	SB	GB				32	24	24	8			
8			UI Testing	In Progress	CA									8	6	4
9			Documentation	In Progress	GB									8	8	2
10		Web partification		Complete	GB		8	5	8							
11		AAA Use Cases														
12			ABC Management	Complete	RG	AT	4	4	4	4	4	4	4	4		
13			Personal Information Management	Complete	RG	AT	4	4	4	4	4	4	4	4		
14			Address Management	Complete	RG	AT	36	36	32	24	16	16	12	6		
15			DEF Management (scenario 1)	Complete	RG	AT	8	8	5	5	4	4	4	4		
16																
17																

Individual Hours

CJ	60	57	52	36	36	36	28	21	4	1
SB	52	47	43	60	44	44	28	20	4	1
CC	40	37	32	16	16	16	8	1		
AM	24	20								
GB	64	52	31	40	24	24	8	8	8	2
CA								8	6	4
TV										
RG	52	52	45	37	28	28	24	18		
AT	52	52	45	37	28	28	24	18		
ST										

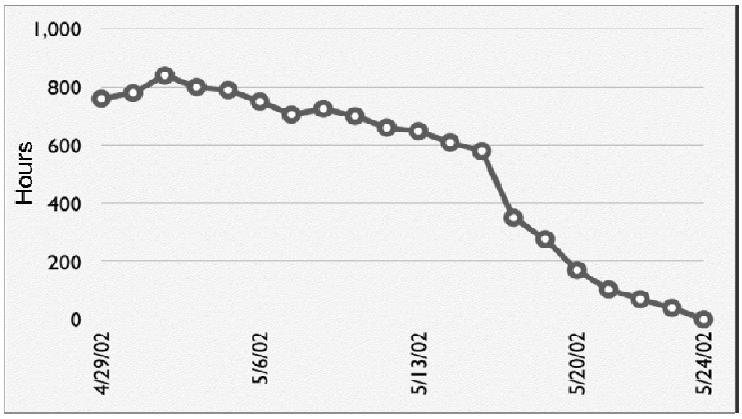
Colours: Not Started, In Progress, Complete, Deferred, Testing

Tasks: Summary of Item, Single Line Item, Support Item

Day	1	2	3	4	5	6	7	8	9	10
Linear	344	323	280	259	212	165	118	74	25	-25
Adjustments				32				16		

UNIVE
I OSL

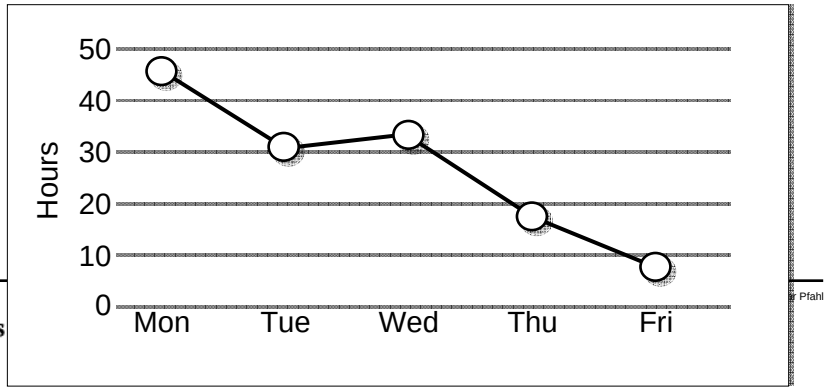
Sprint Burndown Chart – Example



Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

Tasks	Mon	Tues	Wed	Thur	Fri
Code the user interface	8	4	8		
Code the middle tier	16	12	10	7	
Test the middle tier	8	16	16	11	8
Write online help	12				



UNIVERS I OSLO

Scalability of Scrum

Scrum of Scrums of …

- Typical individual team is 7 ± 2 people
  - Scalability comes from teams of teams
- Factors in scaling
  - Type of application
  - Team size
  - Team dispersion
  - Project duration
- Scrum has been used on multiple 500+ person projects (e.g., SAP)

Choosing the Right Process Model

Copyright 2010 © Dietmar Pfahl

Difficult to Choose a Process Model

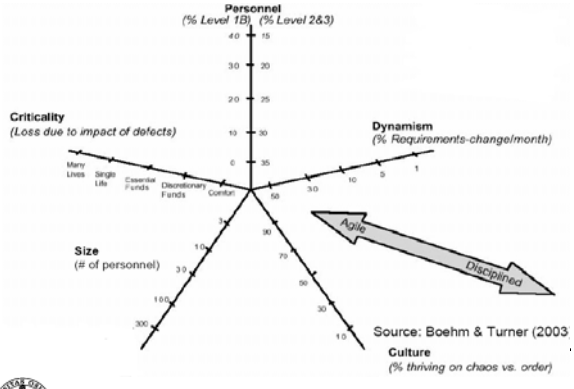
- What you should first decide is whether you actually need a prescriptive process model.
- To make the choice it is important to know your organization/project.
  - What characteristics does the project have?
  - What characteristics affect the choice of the process model?
  - Can we use the same model everywhere, or do we need variants (a repertoire of different models)?

Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

How much *Agility* is Recommended?

- Source: Boehm, B.; Turner, R.; Observations on balancing discipline and agility, Proceedings of the Agile Development Conference, 2003. ADC 2003. Page(s):32 - 39



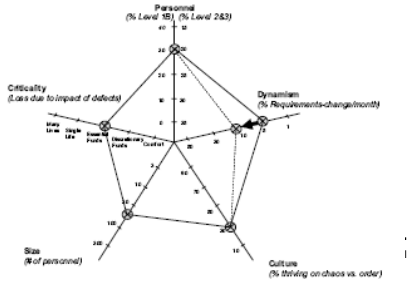


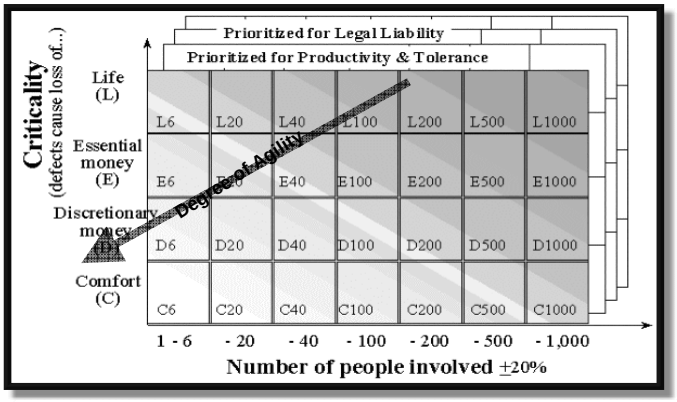
Figure 2. Sample highly-mixed profile

UNIVERSITETET I OSLO

Alistair Cockburn – Project Categorizing

"Any one methodology is likely to be appropriate for only one of the boxes on one of the planes. Thus, at least 150 or so methodologies are needed!"

[Alistair Cockburn: Selecting a Project 's Methodology. IEEE Software 17(4): (2000)]

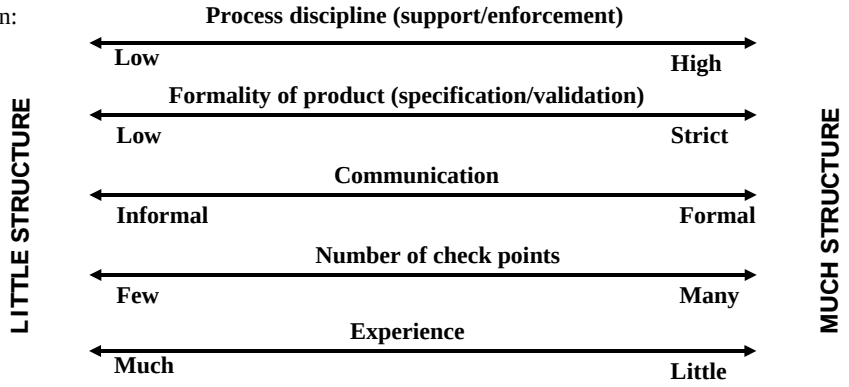


Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

How Much Structure?

Hohmann:

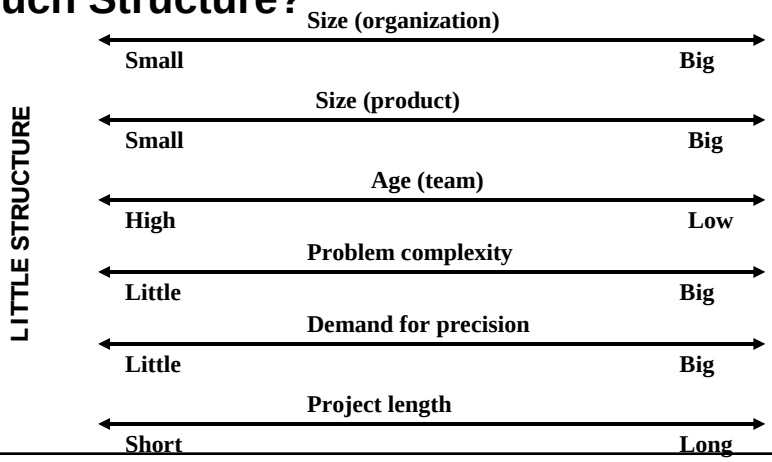


Copyright 2010 © Dietmar Pfahl

UNIVERSITETET I OSLO

How Much Structure?

Hohmann:



Copyright 2010 © Dietmar Pfahl