Planning & Managing Projects

Jesper Andersson

Planning the Project – 101

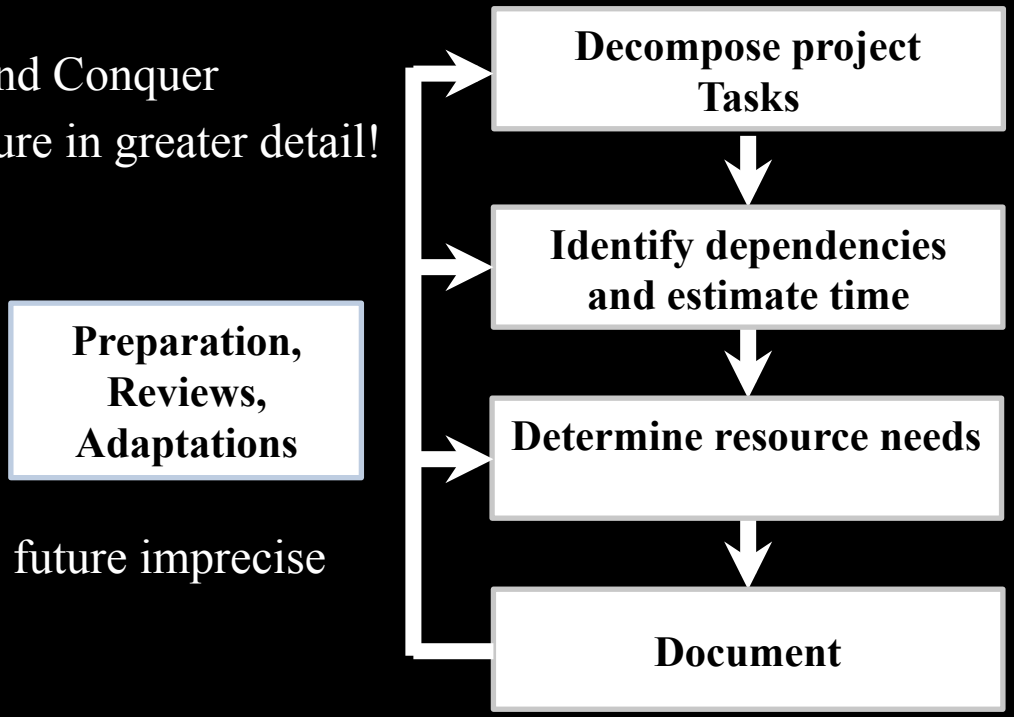
- ✓ Objectives, where do want to end up with this project?
  - These are typically ≈ to the requirements
  - Internal and external objectives

- ✓ Strategies, how will we reach our objectives
- ✓ Plan, allocation of resources and definition of control points along the road

- ✓ Bottom-line is that you need to have a rough understanding of the requirements to plan!

Planning – Solving a complex problem!

- ✓ Divide and Conquer
- ✓ Near-future in greater detail!



- ✓ Keep the future imprecise

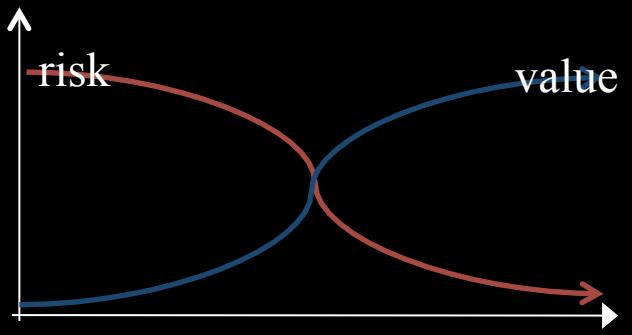
Principles for Planning

- ✓ Short term planning is easier than long-term planning.
  - Unknown risks
  - Resources and Goals
- ✓ “Yesterdays weather is the best forecast for today”
- ✓ Plan with ”slack” – a changed starting date is a changed starting date.

Principles for Planning

- ✓ The Mongolian hoard theory – Assuming that adding more people to a task will speed up its completion. **Will not work!!!**
- ✓ Match competencies and experiences with task characteristics
- ✓ Balance your teams

Software Risk Management



- ✓ Proactive practice for minimizing uncertainty and potential loss associated with a project.
- ✓ Through informed decision making (no guessing game).
- ✓ Performed continually over the life of a system.

- ✓ Important from day 1 to the end of life for the product”

Uncertainty and uncertainty types

- ✓ Complexity and heterogeneity *primary cause* for Uncertainty
- ✓ Uncertainty types
  - Lack of Knowledge – *Facts that are not known, or are known only imprecisely, that are needed to complete the system architecture in a rational way.*
  - Lack of Definition – *Things about the system in question that have not been decided or specified.*
  - Statistically Characterized Variables – *Things that cannot always be known precisely, but which can be statistically characterized, or at least bounded.*
  - Known Unknowns – *Things that it is known are not known*

Uncertainty: Risks/Opportunities and Mitigation

- ✓ Risks
  - Failure
  - Degradation
  - Market shifts
  - Need shifts
- ✓ Mitigations
  - Margins
  - Verification
  - Generality
  - Upgradability

Risks and Risk types

- ✓ **Technology**
  - The database used in the system cannot process as many transactions per second as expected.
  - Software components which should be reused contain defects which limit their functionality.
- ✓ **People**
  - It is impossible to recruit staff with the skills required.
  - Key staff are ill and unavailable at critical times.
  - Required training for staff is not available.
- ✓ **Organisational**
  - The organisation is restructured so that different management are responsible for the project.
  - Organisational financial problems force reductions in the project budget.

Risks and Risk types

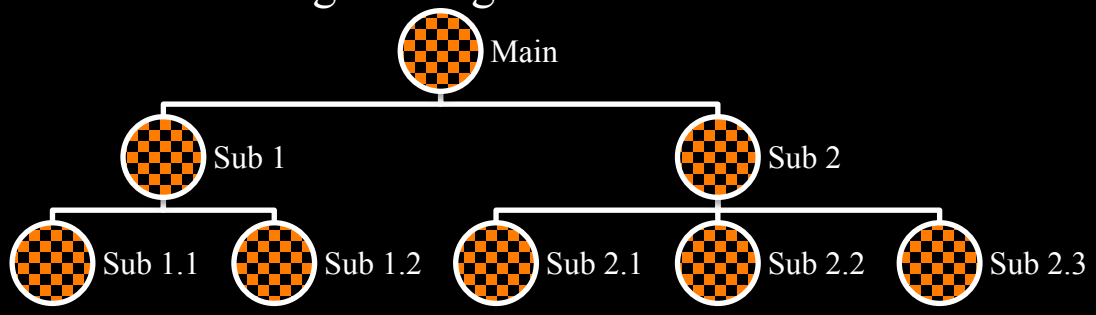
- ✓ **Tools**
  - The code generated by CASE tools is inefficient.
  - CASE tools cannot be integrated.
- ✓ **Requirements**
  - Changes to requirements which require major design rework are proposed.
  - Customers fail to understand the impact of requirements changes.
- ✓ **Estimation**
  - The time required to develop the software is underestimated.
  - The rate of defect repair is underestimated.
  - The size of the software is underestimated.

Two strategies

- ✓ Top-down
  - – Use an existing process model and match it with your project.
  - – Adapt and Instantiate the process!
  - – Allocate resources to process roles that perform activities
  - – Define project specific mile stones
- ✓ Bottom-up – Planning game
  - – Plan ‘releases’ focus on increments/releases
  - – Plan iterations

Work Breakdown Structures (WBS)

- ✓ A hierarchical representation of activities and tasks
- ✓ Start with the most important milestones.
- ✓ Refines milestones into aktivities and tasks with their own mile-stones.
- ✓ Managing the WBS is a continous process, its an ”active document” that will change during the course of the project.



Example

Example

- ✓ Set the Objective
- ✓ Create course grained plan upfront
- ✓ Revise and detail along the way

Set the objectives

- ✓ What type of gathering
  - Casual, formal, kids firendly
- ✓ Guest
- ✓ What type of food
  - Traditional?
  - Something new?
  - Select Recipes
  - Are they doable

Turn Ideas into a Plan

- ✓ Do you have a head count?
- ✓ Now you can finalize the recipes

- ✓ Make shopping list
- ✓ What is the serving plan.
- ✓ Purchase equipment?
- ✓ Décor, candles etc.

Make the schedules

- ✓ Order the turkey
- ✓ Something you can not purchase in your local store

- ✓ Order your items

- ✓ Develop you personal cooking timeline

The First Shopping Trip

- ✓ Start with the Non perishables

- ✓ Any home improvement projects
- ✓ Purchasing towels/soap

- ✓ Clean the freezer
- ✓ Start cooking food now –Bread , Pie dough ,Turkey stock
- ✓ Start working on the decors

Preparations in your home begin

- ✓ Start preparing your home
- ✓ Polish silver
- ✓ Table linens
- ✓ Clean the house

Last week

- ✓ Purchase perishables
- ✓ The day before
  - Do as much of the cooking the day before
  - Defrost
  - Prepare side dishes
  - Set the table

Party time!

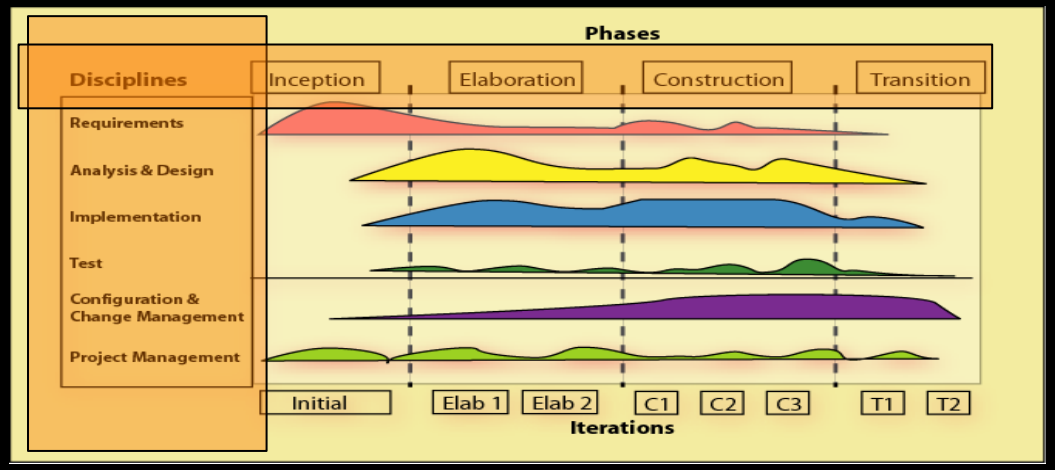
- ✓ Stuff and Roast the turkey
- ✓ Bread
- ✓ Chill Beer and Wine
- ✓ …
- ✓ …

This plan is

- ✓ Based on a generic structure (know how/experience)
- ✓ Tailored for specific needs and wish lists
- ✓ No detailed recipes (designs)

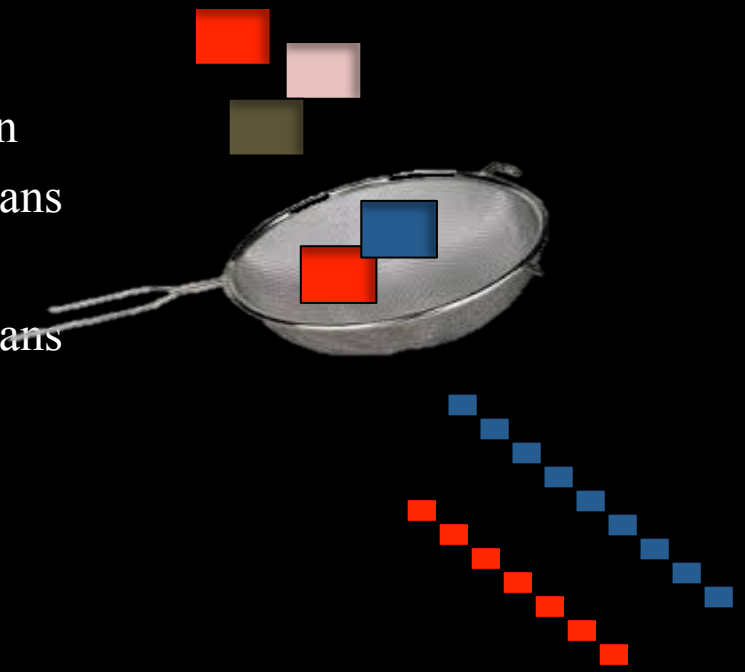
- ✓ Detailed along the way
- ✓ Adopted

UP – Phases

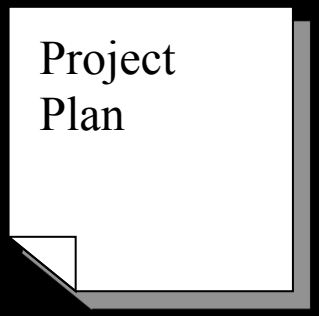

Phases
Disciplines
Inception
Elaboration
Construction
Transition
Requirements
Analysis & Design
Implementation
Test
Configuration & Change Management
Project Management
Initial
Elab 1
Elab 2
C1
C2
C3
T1
T2
Iterations

Open UP - Planning

- ✓ Corse grained
  - Project plan
  - Iteration plans
- ✓ Fine grained
  - Iteration plans
- ✓ Work Items

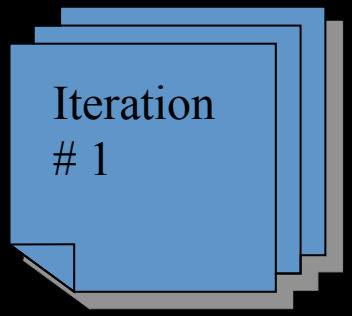


OpenUP planning



- Phase plan (informal)
- Iteration plan (per phase)

Phase



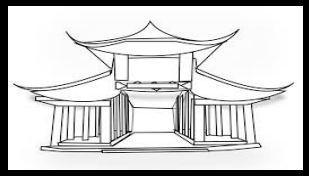
- A time-sequenced set of
  - activities
  - and tasks,
  - with assigned resources,
- containing task dependencies,
- a fine-grained plan.

UP in Short

Inception

Figure out which house the customer wants. Make a plan

Elaboration



Construct the framing, to fix the architecture!

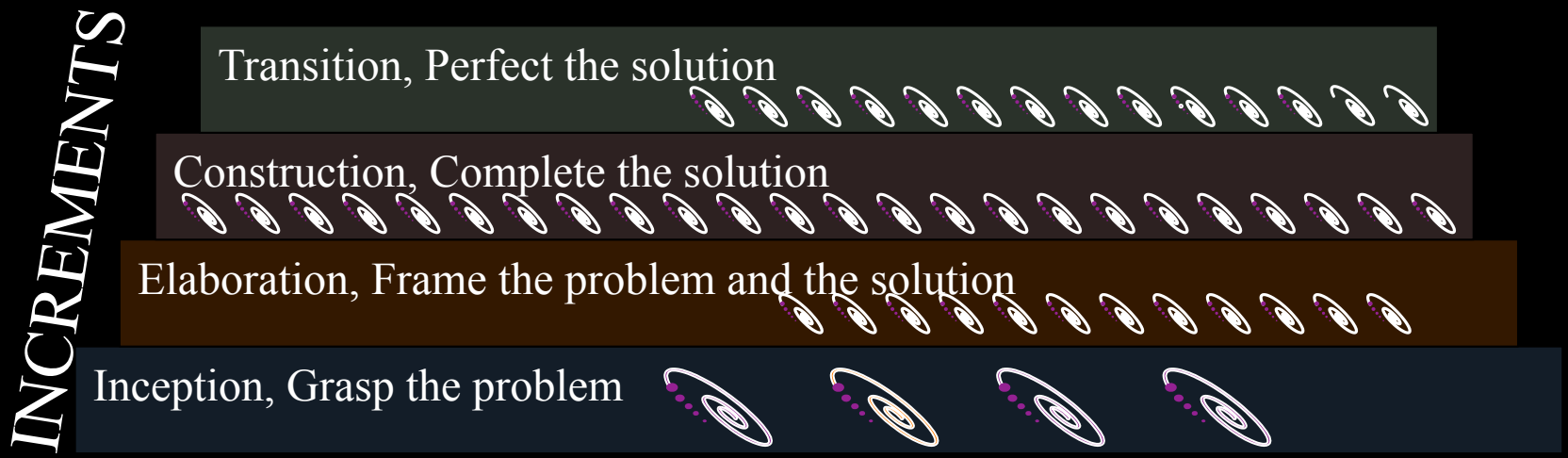
Construction

Complete the house, interior, functions, etc.

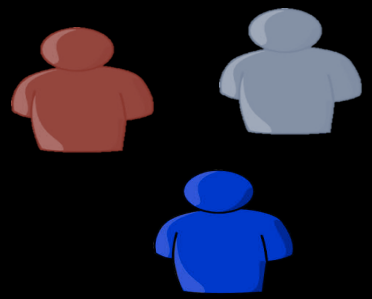
Transition

Let the family live in the house. Use their feedback to improve.

From requirements to Validated solution
Increments and Iterations



An example



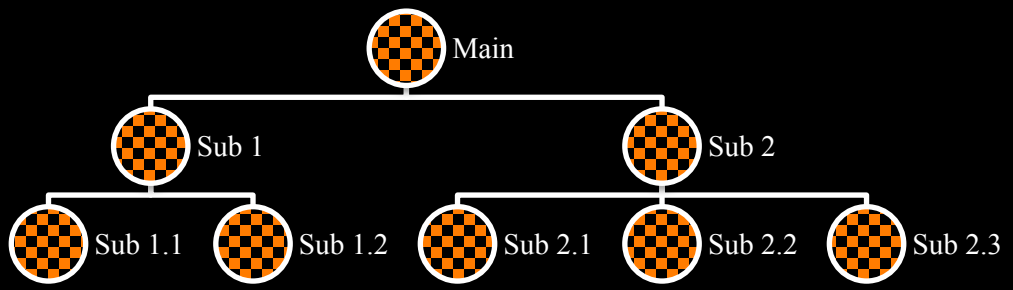
- ✓ Let's study an application where "users share documents"
- ✓ What is most important?
- ✓ Requirements
  - *Functional*, "share files"
  - *Quality*, performance, security, reliability

Course grained Plan – Open UP

- ✓ Inception
- ✓ Elaboration
- ✓ Construction
- ✓ Transition

Understand what to build
Identify key system functionality
Determine at least one possible solution.
Understand the high-level estimate for cost, schedule, and risks.

Can we do this in one iteration?



The Vision Document

- ✓ Defines the stakeholders view of the technical solution
- ✓ Specify it in terms of key needs and features
- ✓ It should contain an outline of the core requirements

- ✓ This will be the objectives – the basis for our plan

<Project Name>
Vision

Usage note: There is procedural guidance within this template that appears in a style named InfoBlue. This style has a hidden font attribute allowing you to toggle whether it is visible or hidden in this template. Use the Word menu Tools→Options→View→Hidden Text checkbox to toggle this setting. A similar option exists for printing Tools→Options→Print.

1. Introduction

2. Positioning

2.1 Problem Statement

[Provide a statement summarizing the problem being solved by this project. The following format may be used:]

The problem of	*[describe the problem]*
affects	*[the stakeholders affected by the problem]*
the impact of which is	*[what is the impact of the problem?]*
a successful solution would be	*[list some key benefits of a successful solution]*

2.2 Product Position Statement

[Provide an overall statement summarizing, at the highest level, the unique position the product intends to fill in the marketplace. The following format may be used:]

For	*[target customer]*
Who	*[statement of the need or opportunity]*
The (product name)	*is a [product category]*
That	*[statement of key benefit; that is, the compelling reason to buy]*
Unlike	*[primary competitive alternative]*
Our product	*[statement of primary differentiation]*

[A product position statement communicates the intent of the application and the importance of the project to all concerned personnel.]

3. Stakeholder Descriptions

3.1 Stakeholder Summary

Name	**Description**	**Responsibilities**
[Name the stakeholder type.]	*[Briefly describe the stakeholder.]*	*[Summarize the stakeholder's key responsibilities with regard to the system being developed; that is, their interest as a stakeholder. For example, this stakeholder:* *ensures that the system will be maintainable* *ensures that there will be a market demand for the product's features*

Pros and Cons with Top-down planning

- ✓ Pros.
  - Adds structure
  - Inexperienced teams get some more guidance
- ✓ Cons.
  - Not natural to re-plan
  - Authoritative

Bottom-up (more agile)

- ✓ Agile principles
  - Self-organizing teams
  - Working software over documentation
  - Customer interactions over contract negotiation
  - React to change instead of following a plan
- ✓ Two levels
  - Release planning (with customers) **Requirements!**
  - Iterations planning (with developers)

Release planning

- ✓ Define ”releases” as mile stones
- ✓ Part of ”scoping the project”
- ✓ Together with your customer
- ✓ Prepare requirements (user stories) that will be included in future releases

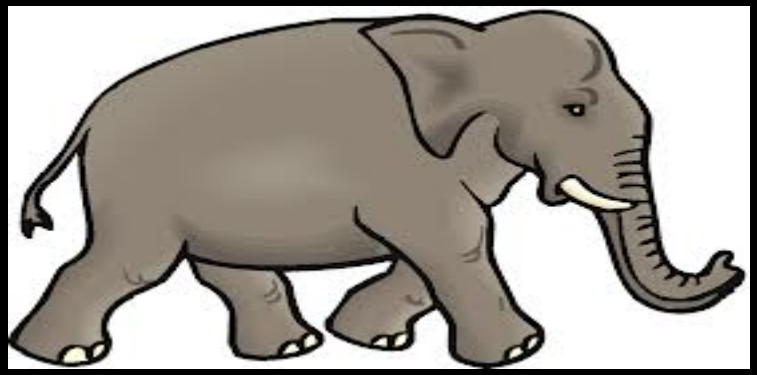
Release Planning

- ✓ Phases
  - Explorative phase: The customer provides a list with stories with great value.
  - Commitment phase: Commit to which functionality that go into the next release and the release date.
  - Execution phase: Adjust the plan when ever needed. For instance, add modify, remove requirements

Iteration planning

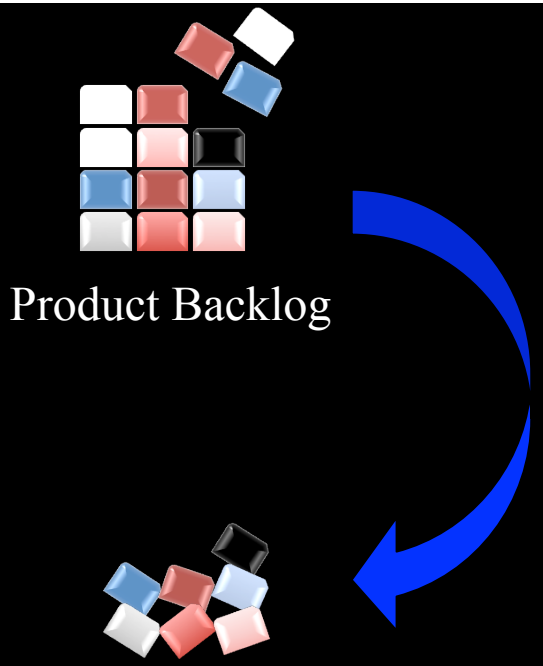
- ✓ Plan the development activities and tasks.
- ✓ Estimate resources.
- ✓ Allocate resources to the tasks
- ✓ Phases
  - Explorative phase: ”Translate” requirements to tasks.
  - Commitment phase: Estimate when tasks should be completed and distribute them among developers.
  - Governing phase: Execute the tasks and compare actual results with planned result.

Planning



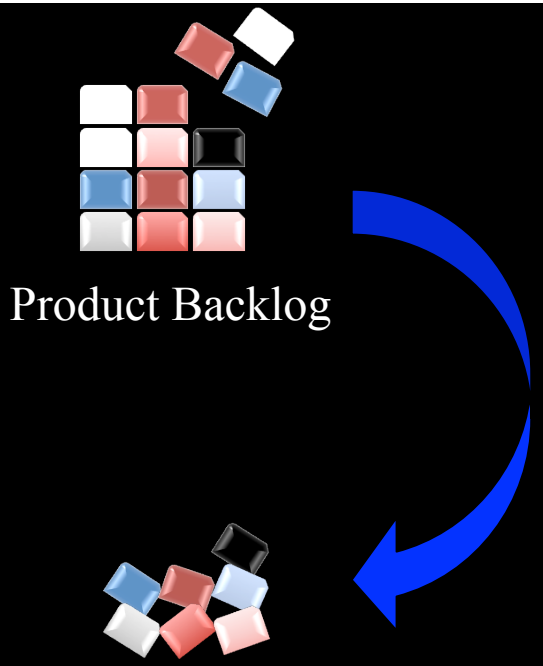
Example - SCRUM

- ✓ 1st step – with the product owner
  - Based on the Prioritized Product backlog
  - Decide which goals you have for the sprint.
  - Create a sprint backlog
  - Participants in the 1st step are
    - Product Owner, Scrum Master,
    - Development team
    - Development team



Example - SCRUM

- ✓ 2nd step – development team internal
  - – Detailed plan for the sprint
  - – Work items
  - – Participants
    - • Scrum Master
    - • Development team



Product Backlog

- ✓ A list of all desired work on the project
- ✓ Contains a combination of
  - "story"- based work, cmp. to. requirements ("let the user search the product database")
  - Task based work ("improve system logging functionality")
- ✓ Prioritized by the Product Owner

- ✓ May also include "Shores" (Swedish:sysslor)
  - "Install tools"
  - Backup the development database

Example on a Product Backlog

Prioritized

Short descriptions

Effort

	Item #	Description	Est	By
Very High				
	1	**Finish database versioning**	16	KH
	2	**Get rid of unneeded shared Java in database**	8	KH
	-	**Add licensing**	-	-
	3	Concurrent user licensing	16	TG
	4	Demo / Eval licensing	16	TG
		Analysis Manager		
	5	File formats we support are out of date	160	TG
	6	Round-trip Analyses	250	MC
High				
	-	**Enforce unique names**	-	-
	7	In main application	24	KH
	8	In import	24	AM
	-	**Admin Program**	-	-
	9	Delete users	4	JM
	-	**Analysis Manager**	-	-
	10	When items are removed from an analysis, they should show up again in the pick list in lower 1/2 of the analysis tab	8	TG
	-	**Query**	-	-
	11	Support for wildcards when searching	16	T&A
	12	Sorting of number attributes to handle negative numbers	16	T&A
	13	Horizontal scrolling	12	T&A
	-	**Population Genetics**	-	-
	14	Frequency Manager	400	T&M
	15	Query Tool	400	T&M
	16	Additional Editors (which ones)	240	T&M
	17	Study Variable Manager	240	T&M
	18	Haplotypes	320	T&M
	19	**Add icons for v1.1 or 2.0**	-	-
	-	**Pedigree Manager**	-	-
	20	Validate Derived kindred	4	KH
Medium				
	-	**Explorer**	-	-
	21	Launch tab synchronization (only show queries/analyses for logged in users)	8	T&A
	22	Delete settings (?)	4	T&A

User stories

- ✓ “User stories” short simple descriptions of functionality.
- ✓ The ”story” uses the the perspective of the individual that asks for the function.

As a <type of user>, I want <some goal> so that <some reason>

- ✓ User stories can be written down on for instance ”sticky notes”, which simplifies prioritization and planning work
- ✓ It is not comparable to a traditional requirements specification
- ✓ ”Incomplete” requirements until you start to discuss them.

Example

- ✓ **As a** club **we like to** register a team **so that** it may participate in a contest

- ✓ Who?
- ✓ What?
- ✓ Why?

- ✓ How can we add more details?
  - Divide in to more, smaller and more detailed stories.
  - Add conditions. ”We are satisfied when…”

From Goals to Sprint Backlog

- ✓ The development team transforms the sprint goals into tasks.
  - The goal is a selection of user stories in the product backlog
  - The selection is made by the development team
- ✓ The team self-organizes to complete the tasks
- ✓ The backlog evolves (incremental)

Sprint Backlog during a Sprint

- ✓ Changes
  - team members are expected to update the sprint backlog at least once every day
  - **Remember!** Only the team can make the changes
- ✓ Modify the estimated work remaining

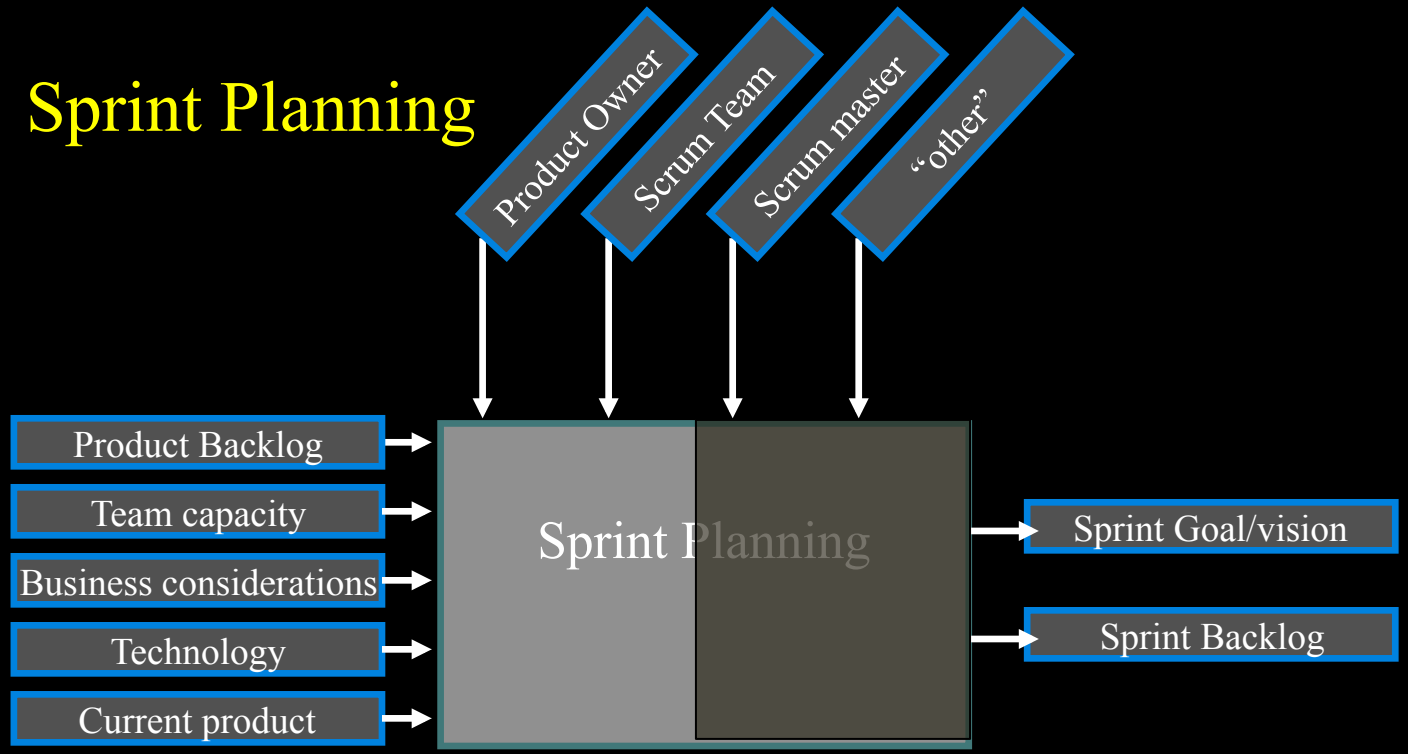
- ✓ Avoid long lists
- ✓ Divide big tasks

- ✓ Each task has its own status

Sample Sprint Backlog

	Days Left in Sprint	15	13	10	8	
Who	Description	7/22/2002	7/24/2002	7/26/2002	7/31/2002	
	Total Estimated Hours:	**554**	**458**	**362**	**270**	**0**
-	**User's Guide**	-	-	-	-	-
SM	Start on Study Variable chapter first draft	16	16	16	16	
SM	Import chapter first draft	40	24	6	6	
SM	Export chapter first draft	24	24	24	6	
	Misc. Small Bugs					
JM	Fix connection leak	40				
JM	Delete queries	8	8			
JM	Delete analysis	8	8			
TG	Fix tear-off messaging bug	8	8			
JM	View pedigree for kindred column in a result set	2	2	2	2	
AM	Derived kindred validation	8				
	Environment					
TG	Install CVS	16	16			
TBD	Move code into CVS	40	40	40	40	
TBD	Move to JDK 1.4	8	8	8	8	
	Database					
KH	Killing Oracle sessions	8	8	8	8	
KH	Finish 2.206 database patch	8	2			
KH	Make a 2.207 database patch	8	8	8	8	
KH	Figure out why 461 indexes are created	4				

Sprint Planning



Planning principles

- ✓ The development team makes the decisions in steps 1 and 2
- ✓ The Product Owner or Scrum Master should not influence
- ✓ After a number of sprints the team will know their "capacity"
- ✓ This capacity is referred to as the "rhythm"

Planning principles

- ✓ In order to plan and measure the team rhythm, we need some “unit of size”. In Scrum you typically use
  - story points
  - “perfect days”
- ✓ Important with all preconditions known
  - Hence no changes during a sprint
  - Scrum masters responsibility

Planning poker

- ✓ Technique for estimating the effort for a task
- ✓ Uses a group's consensus

- ✓ For each task (for example a user story)
  - Every member of the team estimates the effort
  - Pick a card and shows it
  - If some members show different estimates this should be discussed until consensus is reached.

Planning poker - Example

- ✓ Prepare a fruit salad, Maximum time is 30 minutes (sprint duration)
- ✓ ”Minimal Viable Product”
- ✓ Which fruits will be included in the salad?
  - Prioritize
  - Difficulty

Daily Scrum work- Item planning

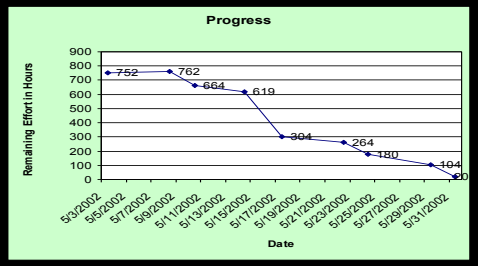
- ✓ Parameters
  - 15-minutes daily, “stand-up”
  - Not a problem solving session
- ✓ Three questions:
  - What did you do yesterday?
  - What will you do today?
  - Which impediments could stop you?

Sprint Review Meeting

- ✓ The development team presents/demonstrates what they have achieved
- ✓ Most common is the informal demo,"2hour preparation time"
- ✓ Feedback to the product owner
- ✓ Participants
  - Customers
  - Product Owner
  - Team
  - Scrum Master

Some more terminology

- ✓ Definition of Done
  - When is the increment completed?
  - The team has the decision
  - Connected to quality not functionality
- ✓ Sprint burn down chart
  - Display effort remaining per day before release
  - Should be zero at the end of a sprint



Managing a Project

- ✓ Frequent feedback
- ✓ Get off on the right foot
- ✓ Keep the momentum

- ✓ Get involved, understand the product
- ✓ Be a leader not a boss

Today's takeaways

- ✓ Top-down
  - Plan ahead,
  - WBS, decompose
- ✓ Bottom up,
  - agile planning
  - the team creates their own plans