# Scrum

# Overview

- The Scrum team.
- Activities during a Sprint.
- Details for stories.
- Recording progress and work still to do.

# Scrum

- Scrum is a framework for the **management** of iterative and incremental product development.
  - It can be used for non-software products.
- It was invented in Japan in 1986 for development of commercial products to provide increased speed and flexibility.
  - A scrum is used to restart a game of Rugby after a minor infringement.
- It was applied to software projects in the early 90s.
- It is a framework for managing complex projects.
  - The **details** can use other approaches such as XP (see later).

# Scrum Theory

- Experiential learning theory (Learn by doing).
- Knowledge and understanding come from:
    - Planning something.
    - Doing it.
    - Reviewing how it went.
    - Adapting the process to be used in the next iteration.
- Transparency
    - The process used is visible to everyone.
- Inspection
    - Frequently check how the work is going.
- Adaptation
    - Change the process if improvements can be made.

# The Scrum Team

- Product Owner: responsible for the business value of the product (customer).
  - Selects what gets done and when
- Scrum Master: ensures the team is motivated and productive.
  - Removes obstacles.
  - Ensures proper processes are followed.
- Team:
  - Between 3 and 9 people in total, including product owner and scrum master.
  - Developers and Testers.
  - User Experience analyst, if appropriate.

# The Scrum Master

- Not a team leader!  The team is self-organising!
- Works with the Product Owner
  - Helps them define the product backlog (stories)
  - Translates the product owners language into words the team will understand.
- Works with the team members
  - Facilitates the events.
  - Coaching and teaching.
  - Removing impediments.
- Works with the rest of the organisation.
  - Liase with other scrum masters.

# Sprints (Iterations)

- From a forward’s point of view, a game of rugby consists of a series of sprints between scrums.
- Between 1 and 4 weeks, typically 2 weeks.
- They are timeboxed (must end on time).
- The goals, sprint backlog and team are fixed.
- If a story can’t be finished, it is returned to the product backlog.
- If problems come to light during a sprint, the team will try and fix them in the sprint.
- Problems that can’t be fixed in the sprint generate stories to be added to the product backlog.
- An entire sprint can be dedicated to fixing problems.

# Sprints (2)

- Risk management
  - Inspection of the product and process happens at the end of each sprint.
  - The maximum amount of work that could be wasted is one sprint’s worth.
  - This reduces the risk that the project goes down a dead end for a long time.
- Sprints can be cancelled by the product owner before the end of the timebox.
  - New information from the sprint shows it is not worth continuing.

# Scrum Events

- Sprint (iteration) planning at the start:
  - The team and Product Owner agree on what to do in the sprint.
  - The meeting is timeboxed for 4 hours (for a 2 week sprint).
  - The stories to be included are chosen.
  - The product owner then leaves and the work is divided up.
- Daily Scrum: Team stand up meeting at the start of each day.
  - Timeboxed at 15 minutes.  Only the team members speak.
  - What did you do yesterday
  - What will you do today.
  - Are there any impediments standing in your way. The scrum master will try and remove them.

# Scrum Events (2)

- Sprint Review: at end of the sprint.
  - Timeboxed to 2 hours (for 2 week sprint)
  - Product owner identifies what has been done.
  - Team discuss what went well, what not so well.
  - Team demonstrate the work they have done.
  - Product owner discusses the remaining product backlog and likely completion date.
  - Discussion on what to do for the next sprint.

# Scrum Events (3)

- Sprint Retrospective: Look for possible improvements.
  - Timeboxed to 1.5 hours ( for 2 week sprint)
  - How the sprint went: people, relationships, processes and tools.
  - What went well.
  - Develop a plan to implement improvements.

# Iteration Length and Velocity

- Iterations are all the same length, to establish a rhythm.
- The iteration **velocity** will then be the expected number of story points (ideal days) that can be completed in each iteration.
  - An initial estimate of the velocity must be made for the first iteration, but it can be modified for later ones in light of experience.
- The estimated project length in iterations will then be the total number of story points on all the user stories divided by the velocity.
  - The velocity is the same for all iterations.

# Planning an Iteration

- An iteration will consist of a small number of stories.
- The team discuss the stories in priority order.
  - The MoSCoW value on the story card.
  - The customer (product owner) must be there to answer questions.
- Each story is split into individual tasks.
  - This is where detailed design happens.
- The developers pick the tasks that they want to work on.
- They estimate the time needed for each of their tasks.
  - Usually in ideal hours.
  - An ideal day contains 8 ideal hours.
- They make sure they are not overcommitted.

# Tasks

- A story is a single action from the user's point of view but will usually involve several objects.
  - Object oriented programming type of objects.
  - This can be found out with a sequence diagram.
- Normally each class is owned by a member of the development team.
  - New user functionality is provided by writing additional methods.
- Each new method will be a different task.

- Sequence diagrams and class structure diagrams will be covered next semester.

# Monitoring Velocity

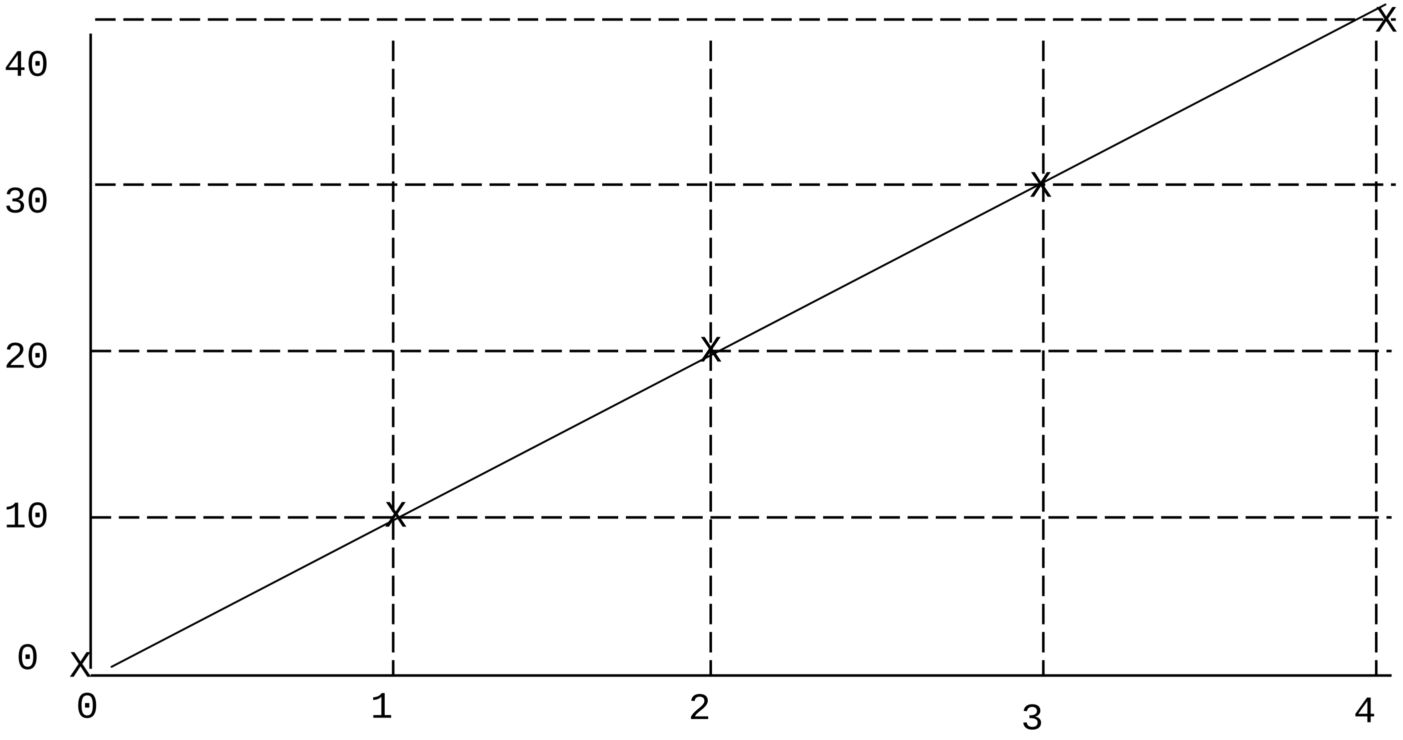
- It is useful to measure predicted and actual effort expended on each story and the iteration as a whole each time an iteration is completed.
  - Actual velocity is calculated by only measuring completed stories.
  - This means that there is no incentive to have many nearly finished stories at the end of an iteration.
- Each story will have two numbers, predicted and actual effort.
- Only use the predicted effort to calculate actual velocity.
  - Encourage accurate predictions.
  - Don’t reward taking too long!

# Story Points Completed Graph

- This is similar to the value added charts in other methodologies.
- The x-axis measures the iteration numbers.
- The y-axis measures story points completed, with two sets of points: predicted and actual.
  - We can draw a horizontal line at the top, the total number of story points for all the stories.
  - We can work this out at the start of the project.
- Additional stories and story points can be added during the project.
- The actual points and number of iterations for the predicted and actual charts can be different.
- New stories don’t appear on this chart, so ...

# Example Ideal Story Completed Graph

- Our target is 40 points in 4 iterations.
- Velocity is 10.



# Iteration Burndown Chart

- This is similar to the story points completed chart but ...
- It measures the work still to be done rather than the work already done.
- Both predicted and actual charts start with the initial estimate of total story points.
- This value is adjusted at the end of each iteration by:
  - Subtracting the story points completed.
  - Adding points for new stories creating.
  - Adjusting for changed estimates of existing stories that have not started.
- The burndown chart can go up at the end of an iteration if a lot more work is discovered while working on it.

# Iteration Hit Rate

- This is simply the fraction of story points completed in the iteration.
  - Total for all completed stories.
- Divided by the planned number of story points for the iteration.
- Expressed as a percentage.
- We plan for 100%, but it will be lower if the iteration runs into problems.

# Planning Releases

- Not every iteration results in a new release.
- Releases can be defined by release date of release functionality.
- We might fix the release date in advance.
  - This will give us a limit on the number of story points that can be achieved.
  - We can then decide what goes into a release, based on this constraint.
- We might want a given functionality for the release.
  - Decide which user stories are needed to achieve this functionality.
  - Add up the story points and decide how many iterations are needed.

# The Release Plan

- Each release will consist of a number of iterations.
- Each iteration will be a pile of story cards.
- Decide which pile a story card goes into depending on priority.
- Make sure each iteration is has roughly the same number of story points.
  - Points on the story cards may not add up exactly to te interation velocity.
  - Example: velocity = 10, story lengths 1,2,2,4, total 9. None of the next stories to work on have length 1.
- High priority stories will be in an early release.

# Example

- A small project has the following 12 stories.

| Story Name | Priority | Effort |
|---|---|---|
| A | Must | 4 |
| B | Must | 2 |
| C | Must | 2 |
| D | Must | 1 |
| E | Must | 8 |
| F | Should | 4 |
| G | Should | 4 |
| H | Should | 2 |
| J | Could | 1 |
| K | Could | 2 |
| L | Would | 4 |
| M | Would | 2 |

# Example (2)

- We estimate our velocity as 10 (2 developers with 2.5 ideal days per week, iteration length 2 weeks).
    - We will try and do everything, and schedule 4 iterations.
    - There will be an initial release after the second iteration and a final release at the end.
- Iteration schedule.

| Iteration | Stories | Total Effort |
|---|---|---|
| 1 | C,E | 10 |
| 2 | A,B,D,H | 9 |
| 3 | F,G,K | 10 |
| 4 | J,L,M | 7 |

# Example (3): Iteration 1

- Iteration 1 goes well and we find that task E is easier than we estimated. It is finished in 6 ideal days.
- Our burndown chart goes down from 36 to 26 because we have completed 10 story points.
  - We use the initial estimate even though we have only expended 8 ideal days of actual work.
- Our hit rate is 10 / 10 or 100%.
  - Initial estimate of completed story lengths / planned effort.

# Example (4): Iteration 2

- This iteration goes badly because story B (2 points) turns out to be a lot harder than we thought.
  - We can’t finish it in this iteration and so add it back to the project backlog.
  - We also discover that it is really 2 stories: B1(2) and B2(4).
  - The release is postponed to after iteration 3.
- We completed 7 points, but added an extra 4 points to the project, so the burndown chart goes down by 3 (7 – 4) to 23.
- Our hit rate is 7/9 or 78%.
- Iteration 3 will now do B1, B2 and F (G, K postponed to 4).
- Iteration 4 will do G, K as well as J, M, but will not do L.

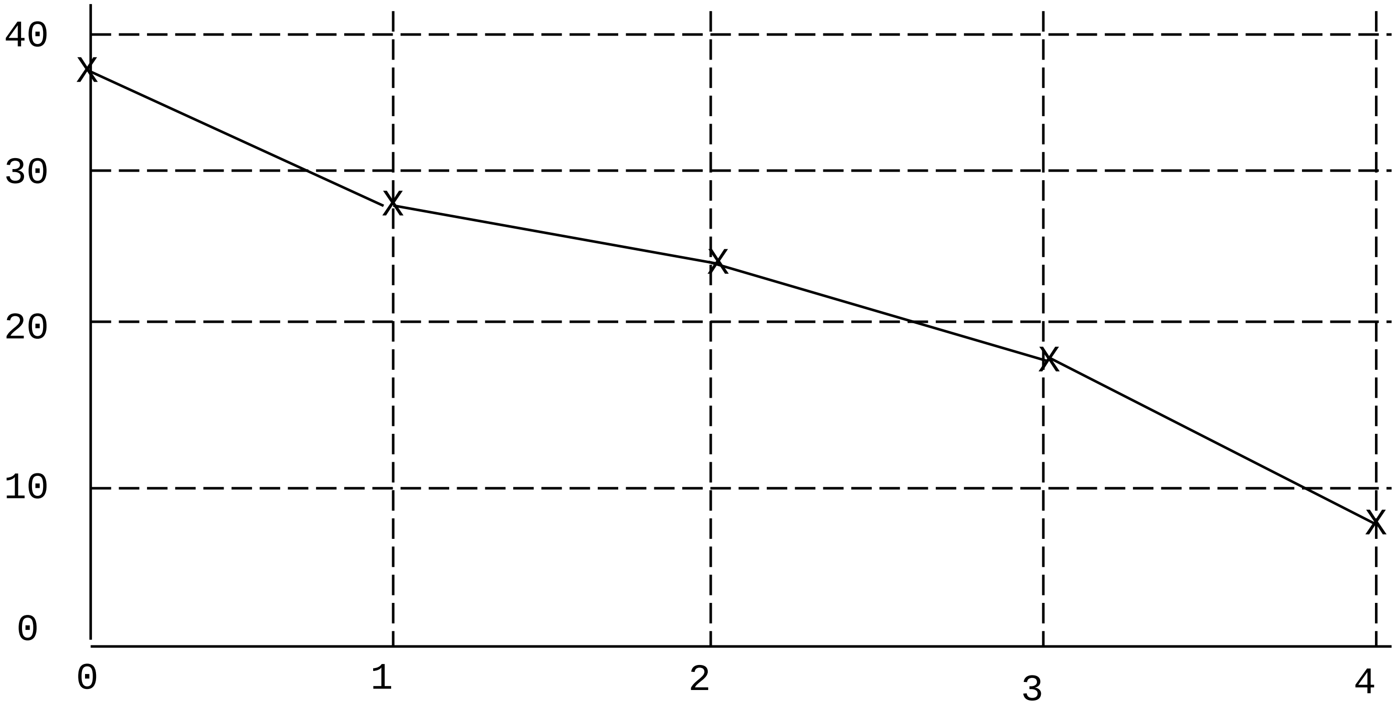
# Example (5): Iterations 3 and 4

- Story B still continues to cause problems
  - B1 takes 3 ideal days
  - B2 takes 5 ideal days.
- As a result, story F has to be pushed back to iteration 4.
  - Iteration 4 no longer plans to do J and M.
- Burndown chart goes down by 6 (B1 and B2 estimates) to 17.
- Hit rate is 6 / 10 or 60%.

- Iteration 4 finishes normally with burndown chart going down by 10 to 7 and hit rate of 100%.

# Example (6): Burndown Chart

- We start with 36 points and end up with 7, since 3 stories worth 7 points were not completed.



# The Lives of Stories

- Stories are discovered in the initial requirements gathering stage and added to the product backlog.
  - Some stories are added later as we learn more about the project.
- They are given a priority.
  - High priority items are refined in more details, perhaps turning into several stories.
  - Low priority items remain at a lower level of detail (epics).
- They are ready to be added to a sprint backlog when:
  - Any prerequisites have been completed.
  - We have a reliable estimate of effort required.
- Estimates can be updated and items re-sequenced as the product is developed.
- They are finished when they are ready to be used by the product owner.
  - Passed all tests.

# Scrum Artefacts

- Product Backlog: Prioritised list of stories.
  - Rough estimates of business value, priority (by the product owner).
  - Rough estimates of effort involved, story points (by the team).
- Sprint Backlog: Stories for this sprint.
  - Stories broken down into tasks.
  - Each task between 4 and 16 ideal hours work.
  - Tasks chosen by team members.
- Burn Down Charts: work remaining on the current sprint, current release and the overall project.
  - Sprint chart updated daily.
- Hit rate: how much of each sprint was completed (percent).

# Scrum Meetings

- The sprint planning meeting lasts half a day.
  - The product owner describes the highest priority items and defines a goal for this sprint.
  - The developers select the items they think they can complete during the sprint.
- The sprint review meeting is at the end of the sprint. No more than 2 hours are allowed for preparation and powerpoint is banned!
- Daily short scrum meetings: what did you do yesterday, what will you do today, are there any obstacles?
  - Standup meetings to make sure they are short.

# Signs of Problems

- A frequent need to revise estimates could mean that the stories are too small.
- There could be difficulties in planning iterations because of dependencies between stories.
  - Try splitting bigger stories differently.
- Gold plating by developers – adding features that were not planned.
  - Avoid making things more complicated than needed.
- Run out of space on the cards could mean too much time spent writing about a story.
  - Could also be caused by thinking too far ahead.

# Signs (2)

- Don’t keep splitting stories to make each iteration have just the right number of story points.
  - Estimates of effort are not that precise.
- If the customer has trouble prioritising.
  - The stories are probably too big.
- The customer won’t write and prioritise stories.
  - Usually a symptom of a blame-based organisation.
  - Find ways for customers and users to express an opinion without the customer taking responsibility if it goes wrong.

# Quiz

1. What does the Product Owner do?  How much are they involved in the day to day development?
2. What does the Scrum Master do?  How do they keep track of what developers do?
3. How are the stories for each iteration chosen?
4. How is work allocated in Scrum?
5. How is progress tracked?
6. How are unexpected difficulties dealt with?

# Reading

- Agile Project Management: Chapter 3.

# Summary

- The Scrum team consists of:
  - Scrum master, who leads rather than manages and solves problems.
  - Product owner, who makes sure the right stories are done.
  - Developers.
- Iterations are called Sprints and have a fixed length and velocity.
  - There is a planning meeting at the start where the work for the sprint is decided.
  - Each day starts with a standup meeting.
  - There is a retrospective at the end of the sprint.

# Summary (2)

- Stories have a priority and estimate of effort required.
  - They start in the product backlog and are then allocated to a sprint and put in a sprint backlog.
- A burndown chart records how much effort is still needed.
  - Updated with the estimated effort of completed stories at the end of each iteration.
- Some sprints result in a product that can be released to the customer.