Software Engineering Project

Morgan Ericsson

morgan@cse.gu.se
@morganericsson
morganericsson
morganericsson

UNIVERSITY OF GOTHENBURG

CHALMERS

eXtreme Programming

eXtreme Programming

- XP is what it says, an extreme way of developing software
  - if a practice is good, then do it all the time
  - if a practice causes problems with project agility, then don't do it

eXtreme Programming

- Team, 3-10 programmers + 1 customer
- Iteration, tested and directly useful code
- Requirements, user story, written on index cards
- Estimate development time per story, priority on value
- Developer starts with discussion with expert user

eXtreme Programming

- Programmers work in pairs
- Unit tests passes at each check-in
- Stand-up meeting daily: Done? Planned? Hinders?
- Iteration review: Well? Improve?
  - Wall list

XP practices

- Whole Team (Customer Team Member, on-site customer)
- Small releases (Short cycles)
- Continuous Integration
- Test-Driven development (Testing)
- Customer tests (Acceptance Tests, Testing)
- Pair Programming

XP practices

- Collective Code Ownership
- Coding standards
- Sustainable Pace (40-hour week)
- The Planning Game
- Simple Design
- Design Improvement (Refactoring)
- Metaphor

Whole Team

- Everybody involved in the project works together as ONE team.
- Everybody on the team works in the same room (open workspace)
- One member of this team is the customer, or the customer representative.

Small Releases

- The software is frequently released and deployed to the customer
- The time for each release is planned ahead and are never allowed to slip. The functionality delivered with the release can however be changed right up to the end
- A typical XP project has a new release every 3 months
- Each release is then divided into 1-2 week iterations

Continuous Integration

- Daily build
  - A working new version of the complete software is released internally every night
- Continuous build
  - A new version of the complete software is built as soon as some functionality is added, removed, or modified

Test-Driven Development

- No single line of code is ever written, without first writing a test that tests it
- All tests are written in a test framework such as JUnit so they become fully automated

Customer Tests

- The customer (or the one representing the customer) writes tests that verifies that the program fulfills his/her needs

Pair Programming

- All program code is written by two programmers working together; a programming pair
- Working in this manner can have a number of positive effects:
  - better code quality
  - fun way of working
  - skills spreading
  - ...

Collective Code Ownership

- All programmers are responsible for all code
- You can change any code you like, and the minute you check in your code somebody else can change it
- You should not take pride in and responsibility for the quality of the code you written yourself but rather for the complete program

Coding standards

- In order to have a code base that is readable and understandable by everybody the team should use the same coding style

Sustainable Pace

- Work pace should be constant throughout the project and at such a level that people do not drain their energy reserves
- Overtime is not allowed two weeks in a row

Simple design

- Never have a more complex design than is needed for the current state of the implementation
- Make design decisions when you have to, not up front

Design Improvement

- Always try to find ways of improving the design
- Since design is not made up front it needs constant attention in order to not end up with a program looking like a snake pit
- Strive for minimal, simple, comprehensive code

Metaphor

- Try to find one or a few metaphors for your program
- The metaphors should aid in communicating design decisions and intends
- The most well known software metaphor is the desktop metaphor

User Stories

- One or more sentences in the everyday or business language
- Captures what a user does or needs to do as part of his or her job function
- Quick way of handling requirements without formalized requirement documents
- Respond faster to rapidly changing real-world requirements

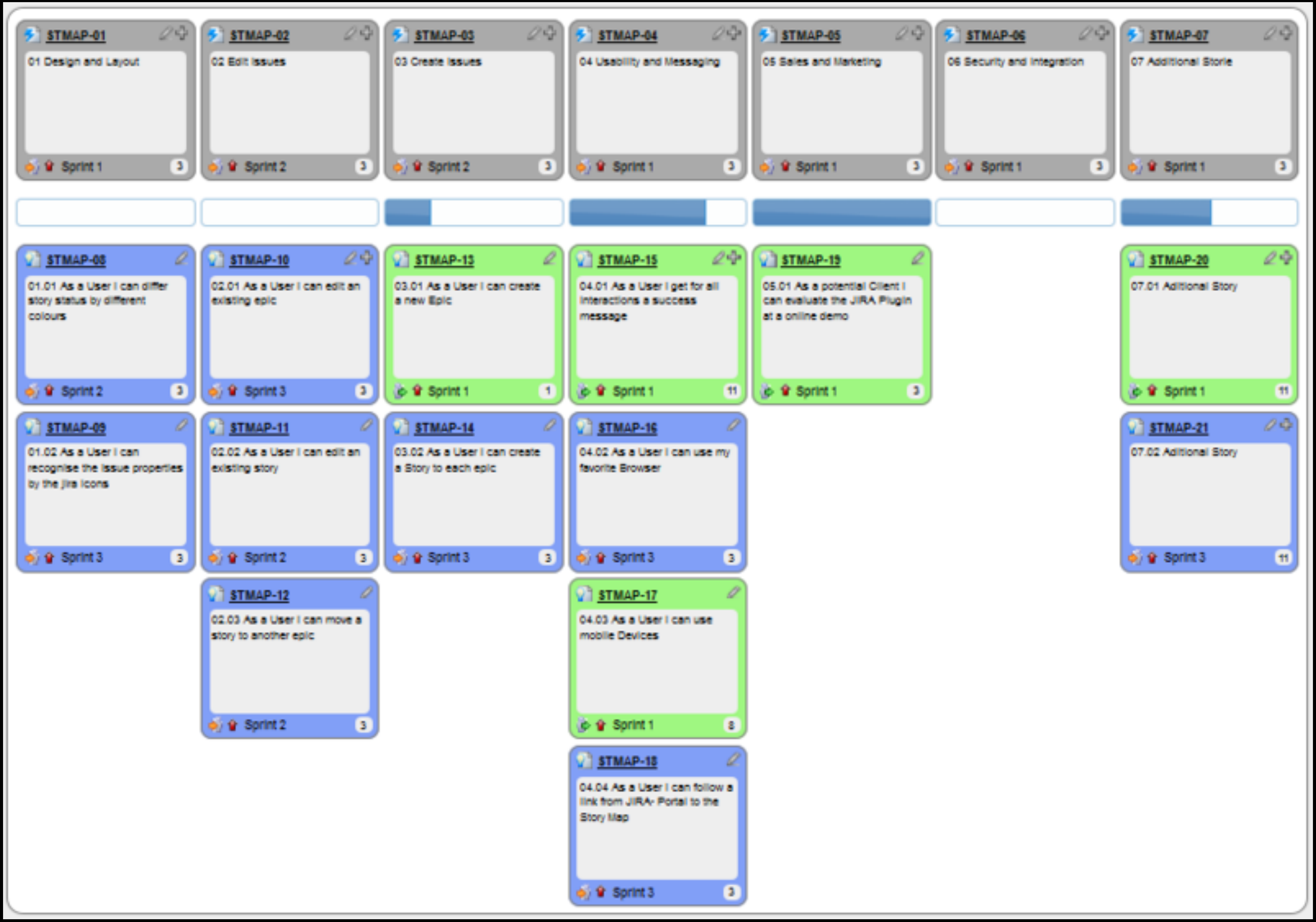
User Stories

- "As a <role>, I want <goal/desire> (so that <benefit>")
- "As <who> <when> <where>, I <what> because <why>."
- *"As a user, I want to search for my customers by their first and last names."*
- *"As a user closing the application, I want to be prompted to save if I have made any change in my data since the last save."*

Benefits

- Represent small chunks of business value that can be implemented in a period of days to weeks
- Needing very little maintenance
- Allowing projects to be broken into small increments
- Being suited to projects where the requirements are volatile or poorly understood. Iterations of discovery drive the refinement process
- Making it easier to estimate development effort
- Require close customer contact throughout the project so that the most valued parts of the software get implemented

Story Maps



STMAP-01 01 Design and Layout Sprint 1 3	STMAP-02 02 Edit Issues Sprint 2 3	STMAP-03 03 Create Issues Sprint 2 3	STMAP-04 04 Usability and Messaging Sprint 1 3	STMAP-05 05 Sales and Marketing Sprint 1 3	STMAP-06 06 Security and Integration Sprint 1 3	STMAP-07 07 Additional Storie Sprint 1 3
STMAP-08 01.01 As a User I can differ story status by different colours Sprint 2 3	STMAP-10 02.01 As a User I can edit an existing epic Sprint 3 3	STMAP-13 03.01 As a User I can create a new Epic Sprint 1 1	STMAP-15 04.01 As a User I get for all interactions a success message Sprint 1 11	STMAP-19 05.01 As a potential Client I can evaluate the JIRA Plugin at a online demo Sprint 1 3		STMAP-20 07.01 Aditional Story Sprint 1 11
STMAP-09 01.02 As a User I can recognise the issue properties by the jira icons Sprint 3 3	STMAP-11 02.02 As a User I can edit an existing story Sprint 2 3	STMAP-14 03.02 As a User I can create a Story to each epic Sprint 3 3	STMAP-16 04.02 As a User I can use my favorite Browser Sprint 3 3			STMAP-21 07.02 Aditional Story Sprint 3 11
	STMAP-12 02.03 As a User I can move a story to another epic Sprint 2 3		STMAP-17 04.03 As a User I can use mobile Devices Sprint 1 8			
			STMAP-18 04.04 As a User I can follow a link from JIRA- Portal to the Story Map Sprint 3 3			

User stories: an example

From *User Stories Applied: For Agile Software Development*

The Project

- South Coast Nautical Supplies sells sailing supplies through a print catalog.
  - GPS, clocks, navigation, weather, life rafts, maps, books
- Wants to move the business online
  - but slow start, focused on small ticket items, such as books

Roles

- **Instructor**: Expected to use the website frequently, often once a week. Through the company's telephone sales group, an Instructor frequently places similar orders (for example, 20 copies of the same book). Proficient with the website but usually somewhat nervous around computers. Interested in getting the best prices. Not interested in reviews or other "frills."
- **Experienced Sailor**: Proficient with computers. Expected to order once or twice per quarter, perhaps more often during the summer. Knowledgeable about sailing but usually only for local regions. Very interested in what other sailors say are the best products and the best places to sail.

Stories for Experienced Sailors

- A user can search for books by author, title, or ISBN
- A user can view detailed information on a book. For example, number of pages, publication date, and a brief description
- A user can put books into a “shopping cart” and buy them when she is done shopping
- A user can remove books from her cart before completing an order

Stories for Experienced Sailors

- To buy a book the user enters her billing address, the shipping address and credit card information
- A user can rate and review books
- A user can establish an account that remembers shipping and billing information
- A user can edit her account information (credit card, shipping address, billing address and so on)
- A user can put books into a "wish list" that is visible to other site visitors

Stories for Instructors

- A user can view a history of all of his past orders.
- A user can easily re-purchase items when viewing past orders.
- The site always tells a shopper what the last 3 (?) items she viewed are and provides links back to them
    - this works even between sessions

Assessment

- "A user can search for books by author, title or ISBN number"
  - what does that mean? Either or, or any possible combination?
- Fix!
  - "A user can do a basic simple search that searches for a word or phrase in both the author and title fields"
  - "A user can search for books by entering values in any combination of author, title and ISBN"

Product vision

Why should you have a product vision?

- It describes the application in general terms
  - including descriptions of the target market, the system users, and the application features
- It assures that everyone is working toward a single objective
- It identifies common goals – a shared gestalt.

What should it contain?

- It captures needs, features, and other common requirements
- It defines problem and the solution on a high abstraction level
- It serves as basis for discussion
  - marketing
  - the development
  - management

What should it contain (cont'd)?

- It captures the essence of the product
  - short, abstract, readable, and manageable form.
- It is a concise description of everything you consider important
  - plain language
  - level of detail that stakeholders understand

For a mid-sized company's marketing and sales departments who need basic CRM functionality, the CRM-Innovator is a Web-based service that provides sales tracking, lead generation, and sales representative support features that improve customer relationships at critical touch points. Unlike other services or package software products, our product provides very capable services at a moderate cost.

http://www.joelonsoftware.com/articles/JimHighsmithonProductVisi.html

For scientists who need to request containers of chemicals, the Chemical Tracking System is an information system that will provide a single point of access to the chemical stockroom and vendors. The system will store the location of every chemical container within the company, the quantity of material remaining in it and the complete history of each container's location and usage. This system will save the company 25% on chemical costs in the first year of use by allowing the company to fully exploit chemicals that are already available within the company, dispose of fewer partially used or expired containers and use a single standard chemical purchasing process. Unlike the current manual ordering processes, our product will generate all reports required to comply with government regulations that require the reporting of chemical usage, storage, and disposal.

Form

- For <target customer>
- Who <statement of the need or opportunity>
- The <product name> is a <product category>
- That <key benefit, compelling reason to buy>
- Unlike <primary competitive alternative>
- Our product <statement of primary differentiation>

Qualities

- Clear and stable
- Broad and engaging
- Short and sweet

Week 3

- **Monday**: Source code management (with Git)
- **Wednesday**: Supervision
- **Friday**: Android development
- **Sunday:** First release (repo + some code)