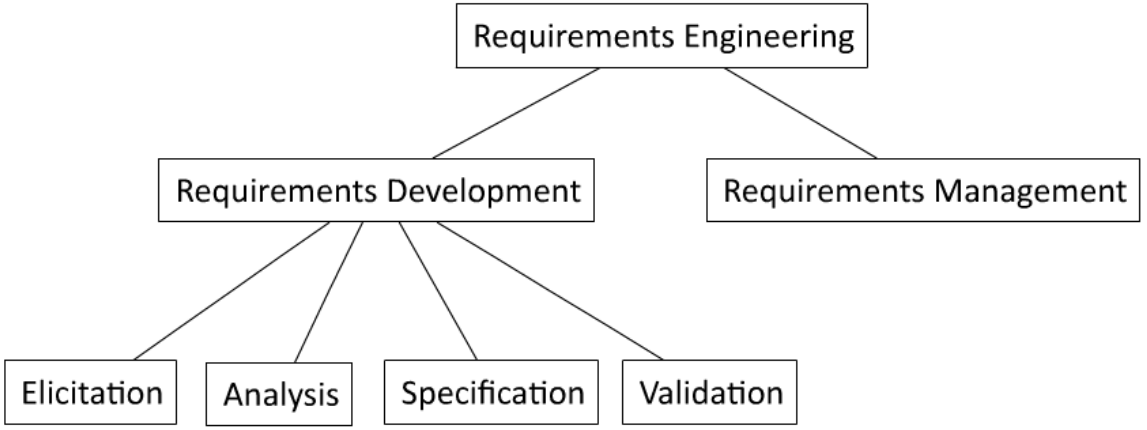
Requirements Engineering

Understanding what you are going to build

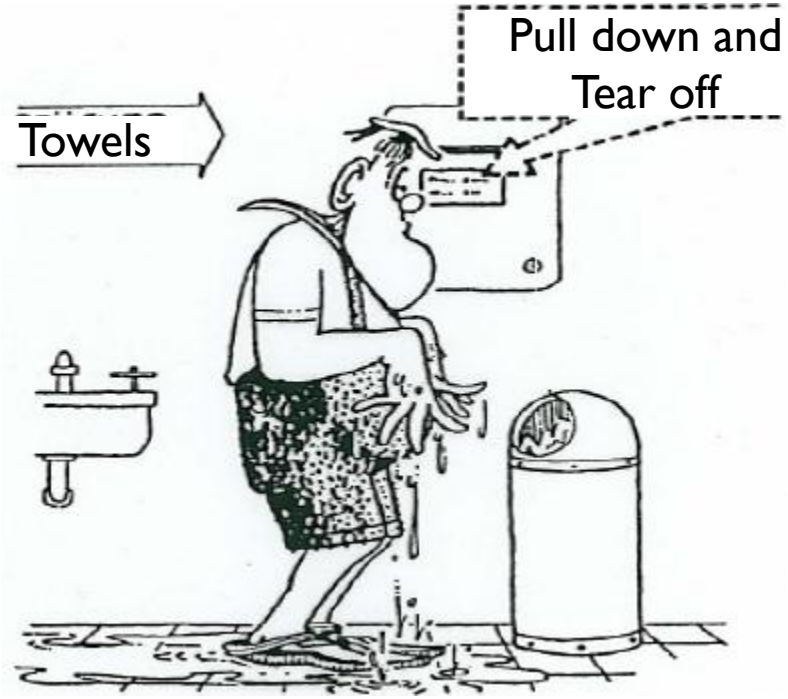
Requirements Engineering

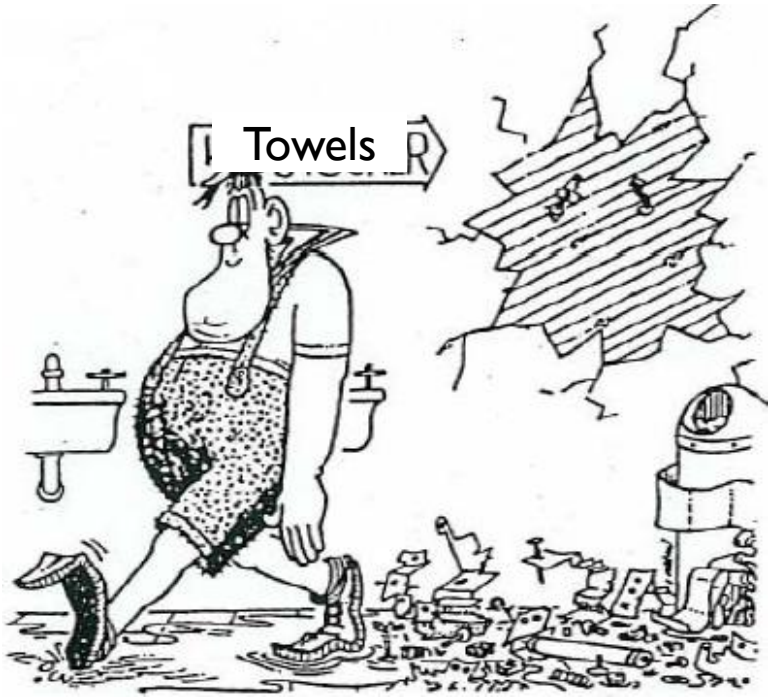
- Requirements Engineering covers all the activities needed in discovering, documenting, maintaining the set of requirements and the constraints of a software system.



Karl Wiegers, Software Requirements, 2nd edition

Different people interpret requirements differently





An Honest Disclaimer

- “We don't claim Interactive EasyFlow is good for anything ... if you think it is, great, but it's up to you to decide. If Interactive EasyFlow doesn't work: tough. If you lose a million because Interactive EasyFlow messes up, it's you that's out of the million, not us. If you don't like this disclaimer: tough. We reserve the right to do the absolute minimum provided by law, up to and including nothing. This is basically the same disclaimer that comes with all software packages, but ours is in plain English and theirs is in legalese. “

- ACM Software Engineering Notes, Vol. 12, No. 3, 1987.

Where Do Requirements Come From?

- Your project stakeholders :
  - direct or indirect users,
  - managers, senior managers,
  - operations staff members,
  - testers,
  - developers working on other systems that integrate or interact with yours,
  - maintenance professionals

Check List for Requirements

- **Validity**: Do they provide what best support the customer's needs?
- **Consistency**: Are there any requirements conflicts?
- **Completeness**: Are all required functions included?
- **Realism:** Can they be done given available budget and technology
- **Verifiability**: Can the requirements be checked?
- **Unambiguous:** They shouldn't be interpreted in more than on way
- **Implementation free**: identify what to do now how to do it
- **Necessary**

Requirements validation techniques

- Requirements reviews
  - Systematic manual analysis of the requirements.
- Prototyping
  - Using an executable model of the system to check requirements.
- Test-case generation
  - Developing tests for requirements to check testability.

Requirements Document

Defines a generic structure for a requirements document that must be instantiated for each specific system.

- Preface
- Introduction
- Glossary
- User requirements definition
- System architecture
- System requirements specification
- System models
- System evolution
- Appendices
- Index

User Stories

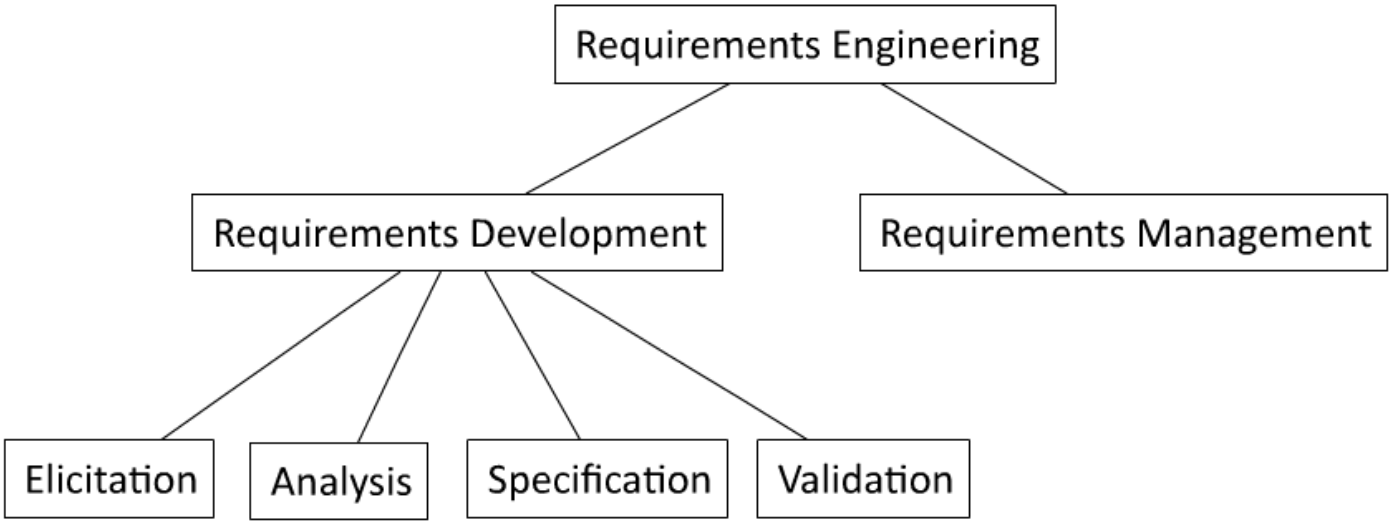
Dr. Fatma Meawad

Bottleneck is Requirements

- Requirements' errors are the greatest source of defects and quality problems
- Most of errors originate in requirements and design activity
- Fixing requirements errors eats up roughly one-third of your project budget
- Requirements are NOT a document:
  - Dialogue

Hooks and Farry 2001

Requirements Engineering

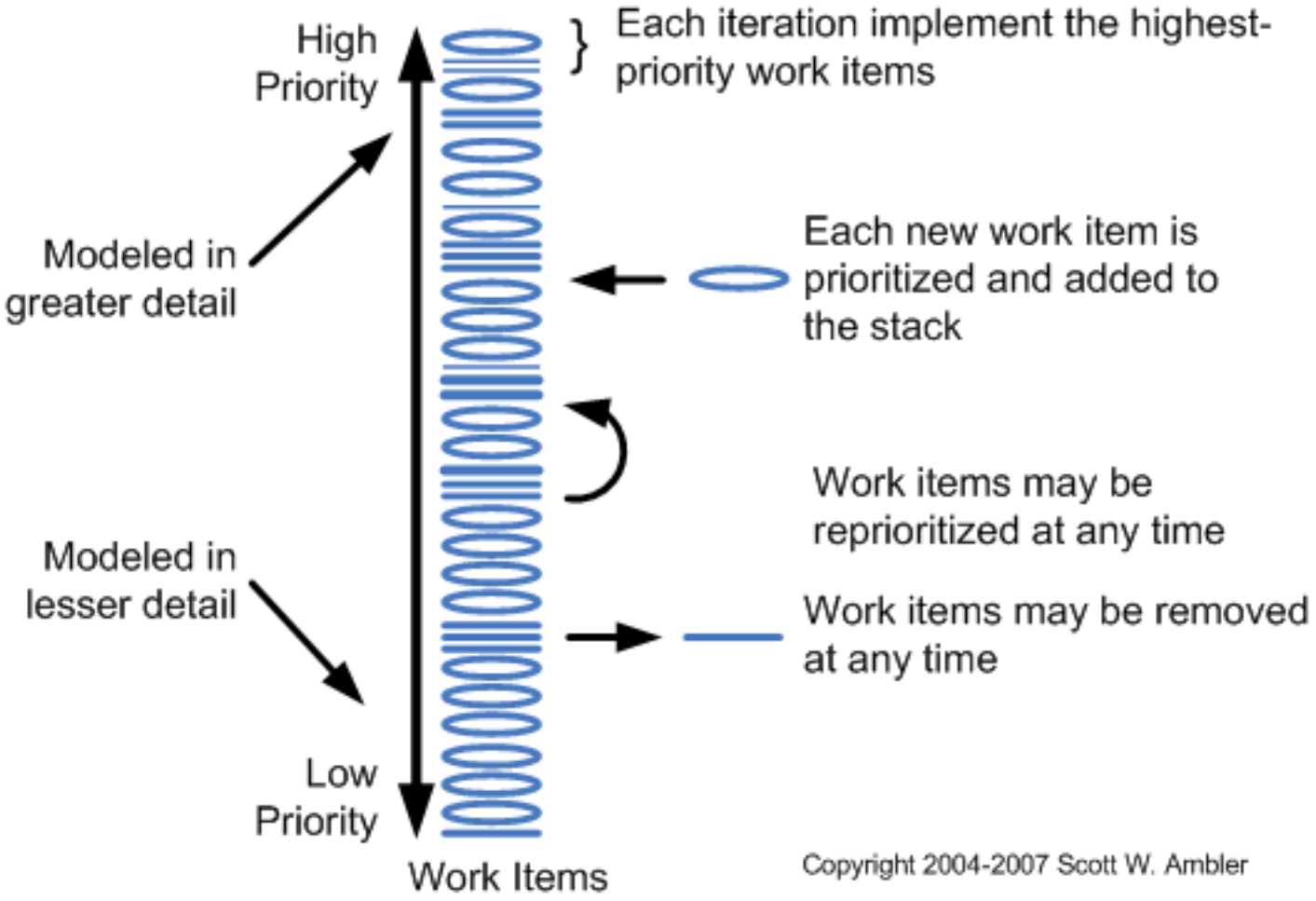


Karl Wiegers, Software Requirements, 2nd edition

Agile Requirements

http://www.agilemodeling.com/artifacts/userStory.htm

Agile Requirements (Change Management)



http://www.agilemodeling.com/artifacts/userStory.htm

User Stories

User stories are simple, clear, brief descriptions of functionality that will be valuable to the user.

- High level requirements
- Should be easy enough for any one to write
- Should act as reminder for a conversation with customers (software requirements is a communication problem)

Informal User stories (Examples)

- Students can purchase monthly parking passes online.
- Parking passes can be paid via credit cards.
- Professors can input student marks.
- Students can order official transcripts.
- Students can only enroll in seminars for which they have prerequisites.

http://www.agilemodeling.com/artifacts/userStory.htm

Formal User Stories

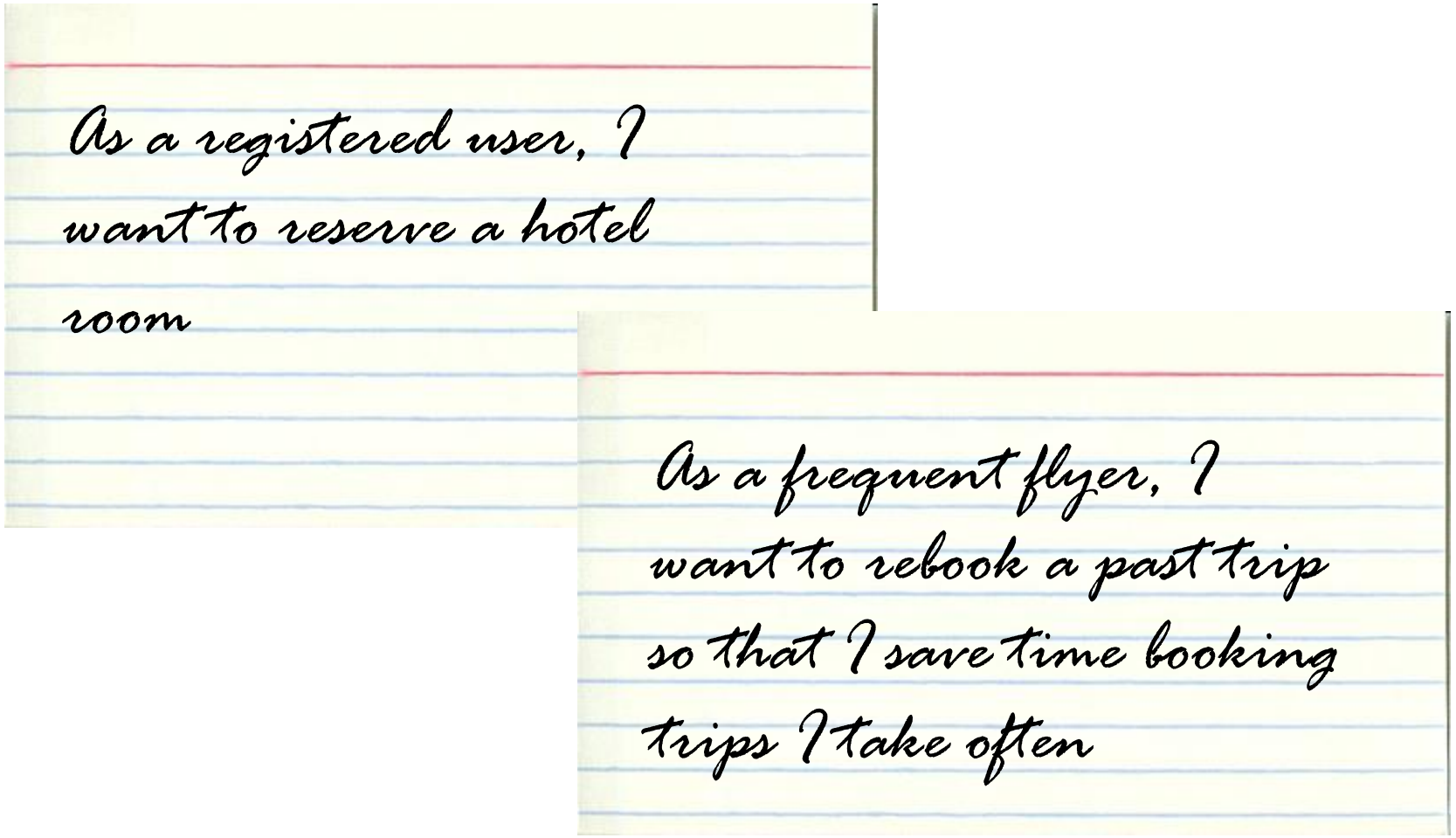
- As a [role], I want [feature] because [reason]
- As a [role], I can [feature]
- As a [role], I can [feature] so that [reason]

- **Why** (reason)
  - gives clarity as to why a feature is useful
  - can influence how a feature should function
  - can give you ideas for other useful features that support the user's goals

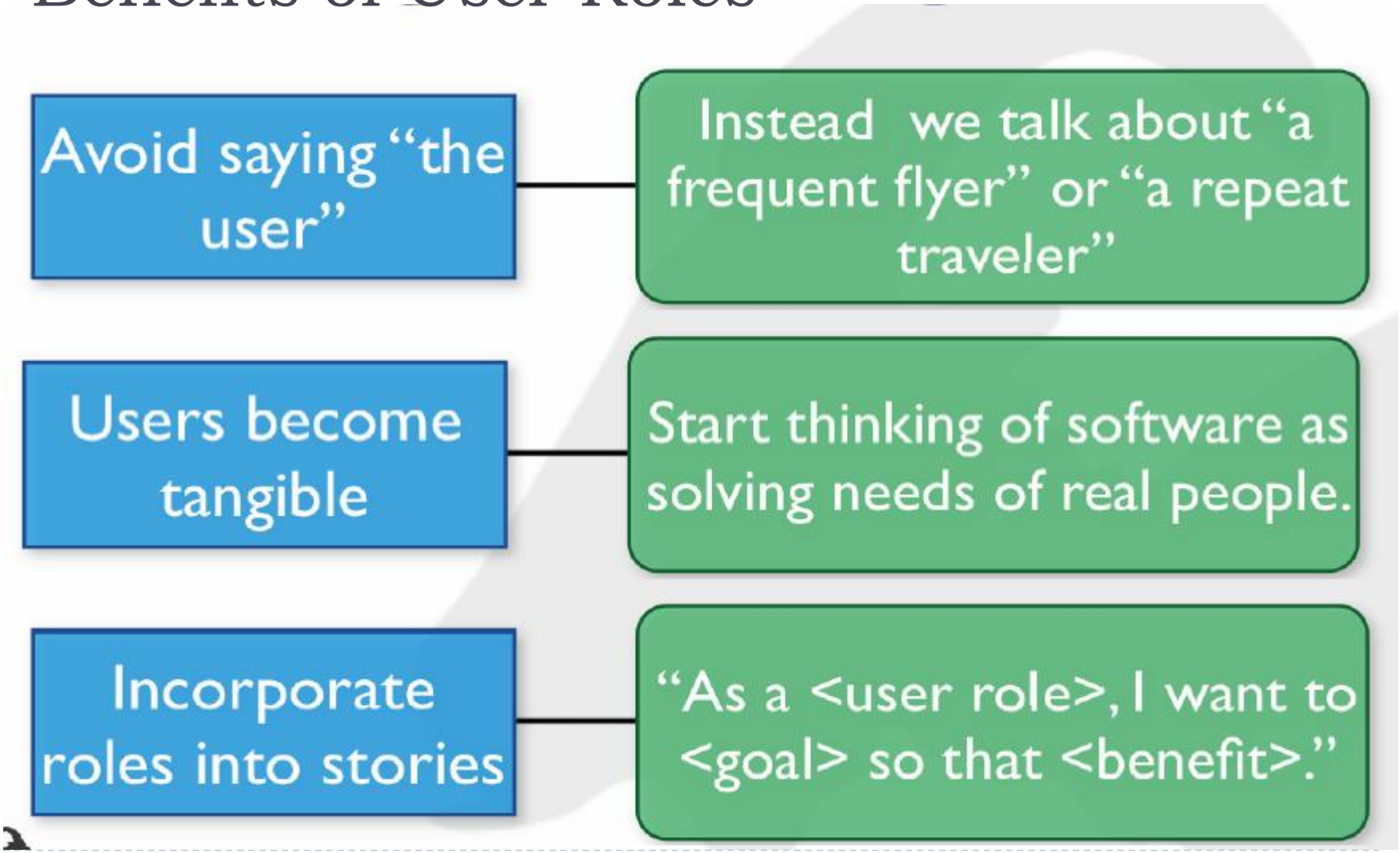
Formal User stories (Example)

- As a [role], I want [feature] because [reason]
- As a registered user I want to log in
  so I can access subscriber-only content

User story cards



Benefits of User Roles



http://www.mountaingoatsoftware.com/system/presentation/file/97/Cohn_SDWest2009_EUS.pdf

Getting details out of a story

We want more details

- As a User, I can cancel a reservation
  - Is the refund to her credit card or is it site credit?
    - Credit card for premium members
    - Site credit for non premium members
  - How far ahead must the reservation be cancelled? & Does the user get a full or partial refund?
    - Premium members can cancel up to last minute without an additional fee
    - Non Premium members can cancel up to 24 hours in advance, otherwise a 10% fee is charged
  - Is that the same for all hotels?
    - Yes
  - For all site visitors? Can frequent travellers cancel later?
    - No
  - Is a confirmation provided to the user? How?
    - Email notification is sent to the user and the airlines admin

Discovering Different Roles

- You might prefer to write it as two different stories

  - As a premium member, I can cancel reservation up to the last minute
    - No fee deduction
    - Email sent to both sides
    - Credit card can be refunded
  - As a non-premium member, I can cancel reservation up to 24 hours in advance
    - 10 % is deducted if the cancelation is less than 24 hours
    - Only site credit is restored
    - Email sent to both sides

Success and Failure Scenario

- What we have just done is derive the conditions to be satisfied for each user story.
- These conditions can be thought of as how your system should behave in case of success or failure
- These scenarios represent to the customer the how to demo section of the User story
- Valuable in your contract with the customer
- They are usually written in the back of the index cards
- Each story has to have a way to test it by a user (acceptance test)

Front and back of a story index card

Front of the Card

- As a registered user, I want to reserve a hotel room

Back of the card

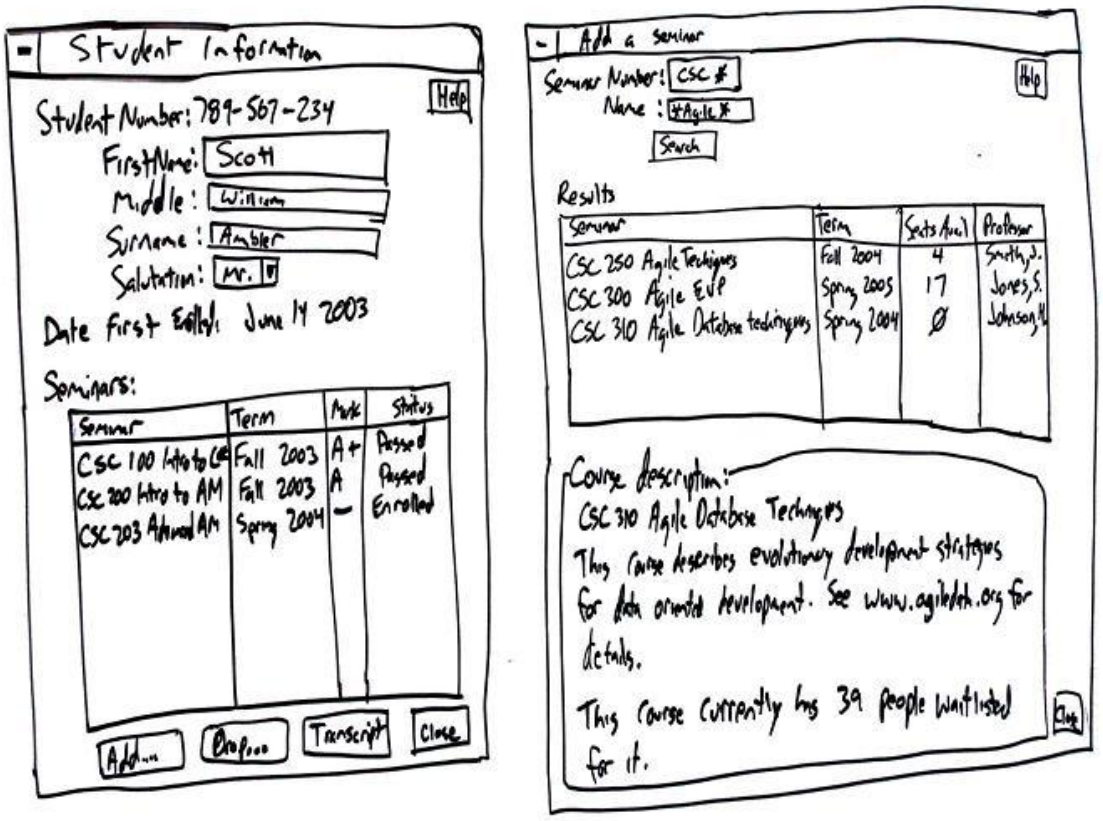
- Verify the user is logged in, otherwise redirect user to login/ registeration page
- Verify the start date is before end date of reservation
- Verify the room is available in the selected duration, otherwise, prompt the user to reselect
- Verify the room is available in the selected duration, otherwise prompt the user with the closest available times
- Verify the user has credit card details in his profile, otherwise redirect to payment portal

Common Mistakes with scenarios

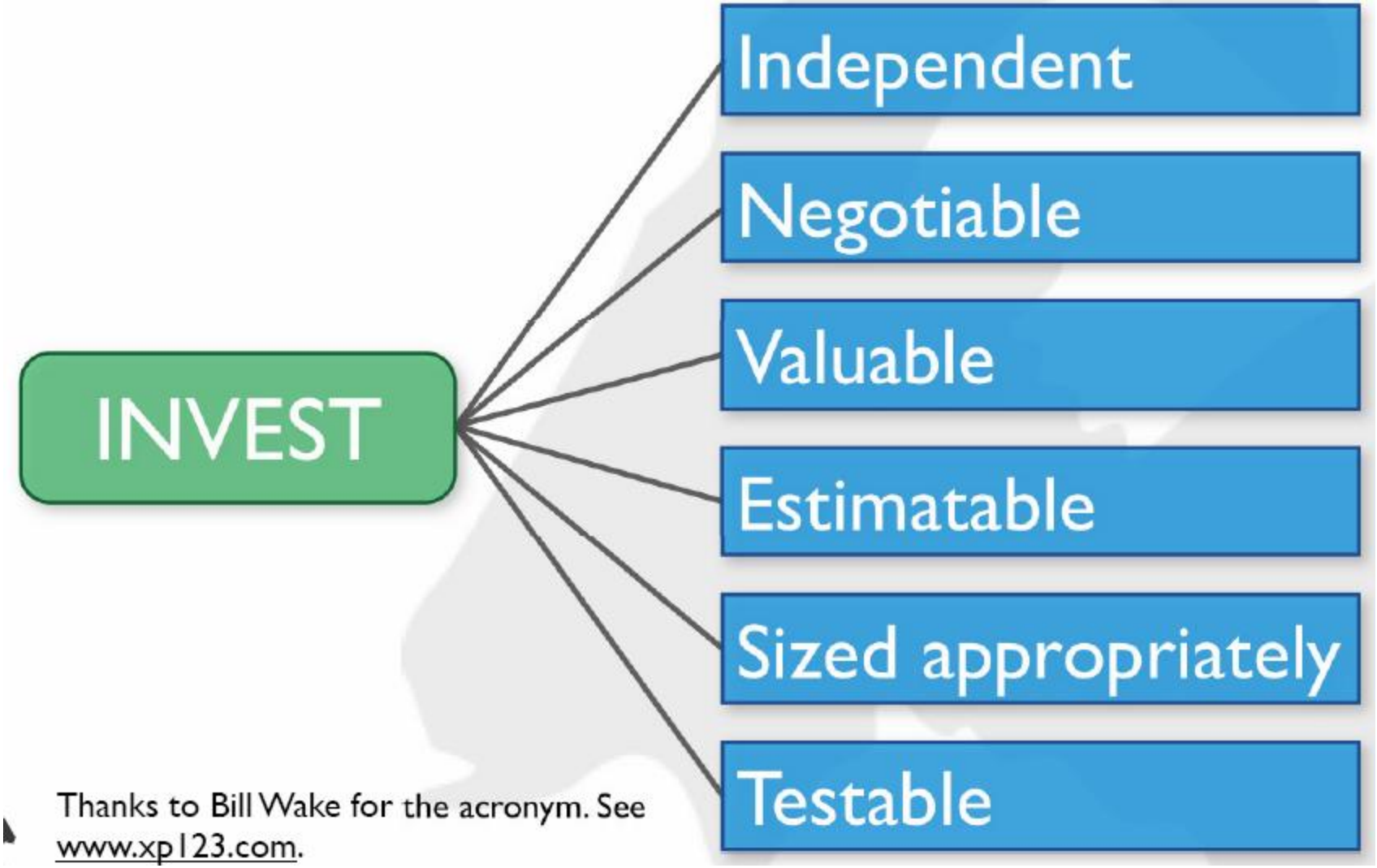
- The details of pre-requisites are ignored
  - Given that everything required is done, then everything is successful. (not acceptable)
  - Given that everything required is not done, then everything is not successful. (not acceptable)
- Failure is not about your code not working, it is about alternate scenarios if all the pre-requisites of a story are not fulfilled
  - User sees an exception???
  - No Failure Scenarios (think well about this)
- The feedback (on success and failure) to the user are ignored
  - For example, redirecting to a specific page, response messages, email, SMS or any kind of notifications sent.

Sketches

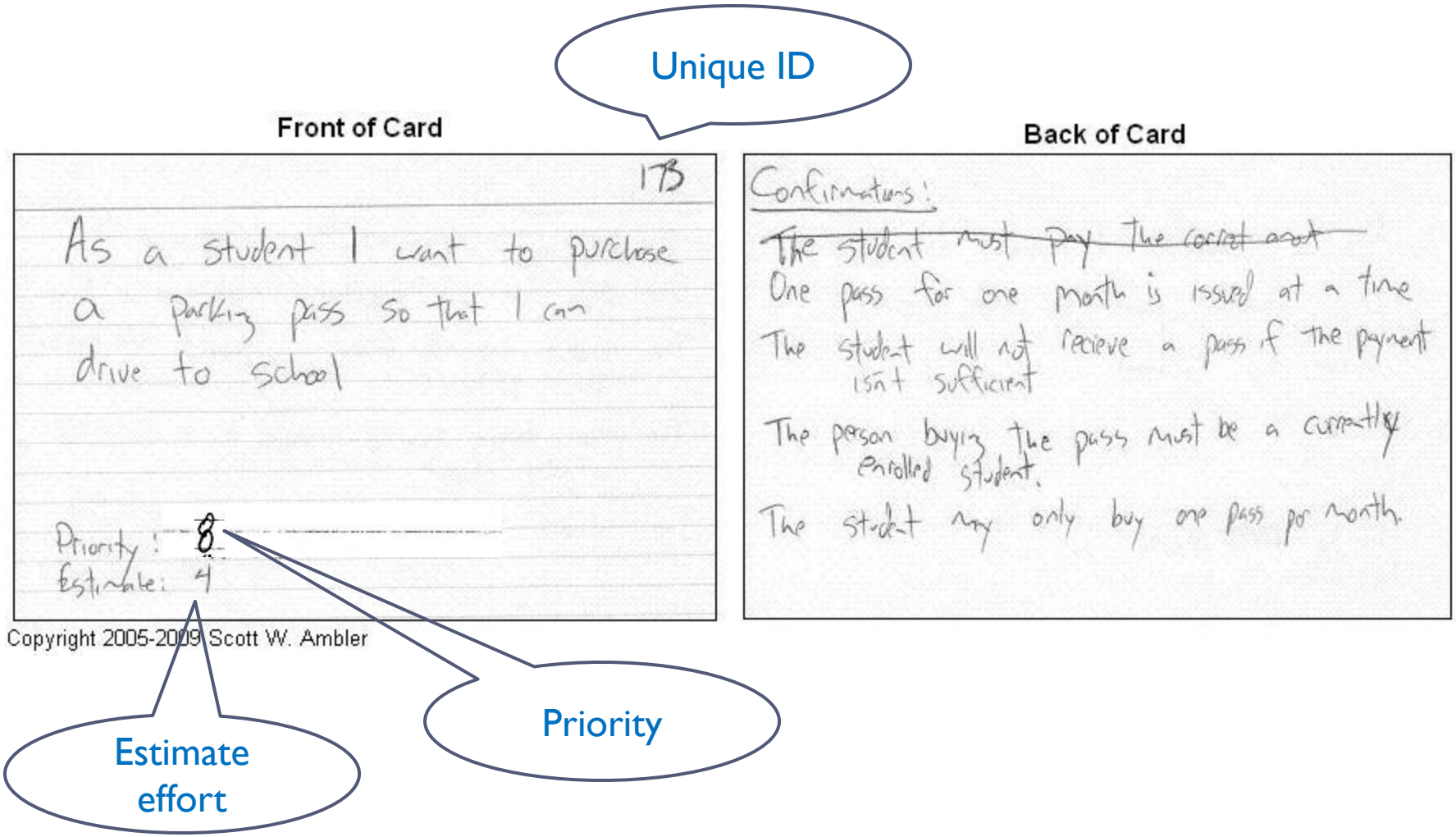
- Sketches can help to get information from the customer
- As a student, I want to enrol in a seminar



What makes a good user story



http://www.mountaingoatsoftware.com/system/presentation/file/97/Cohn_SDWest2009_EUS.pdf



http://www.agilemodeling.com/artifacts/userStory.htm#Detailing

Two main types of requirements:

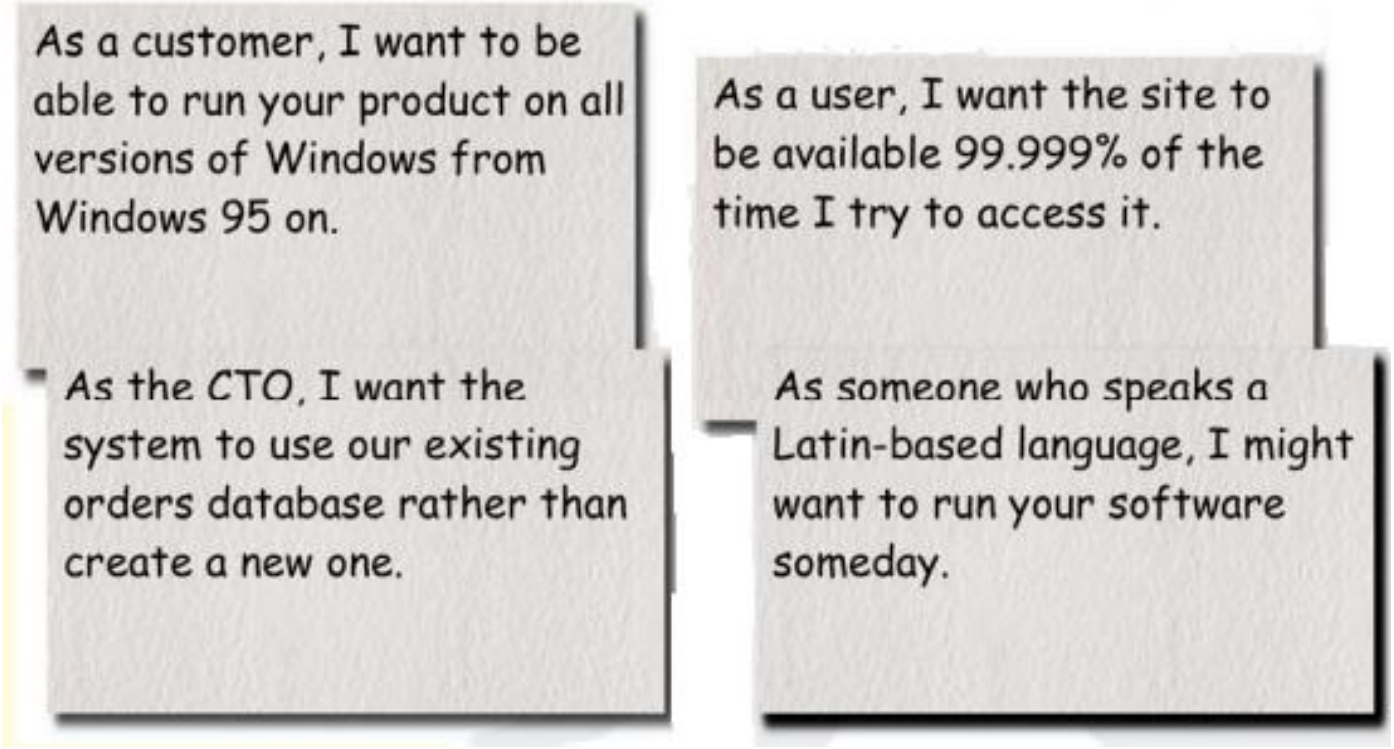
- Functional
- Non Functional

Functional Requirements

- Functions of the system
- Behavioural: How the system should behave in certain situations. How user will interact and use the system.
- A function is described as a set of inputs, the behaviour, and outputs.
- requirements must be clear, correct, unambiguous, specific, and verifiable.

Non Functional Requirements

- Technical aspects that your system must fulfill



Non Functional Requirements

- Constraints on the services or functions offered by the system such as timing constraints, constraints on the development, standards, etc.
  - Usability
  - Performance
  - Reliability
  - Accuracy
  - Security
  - Maintainability

Why User stories

- Stories are understandable
  - Developers and customers understand them
- Move the focus from writing to discussions
- Support and encourage iterative development
- Can easily start with dense stories and disaggregate closer to development time
  - During your User stories writing, you might have a big story. Then try to break it into details. (example next slide)

http://www.mountaingoatsoftware.com/system/presentation/file/97/Cohn_SDWest2009_EUS.pdf

Example

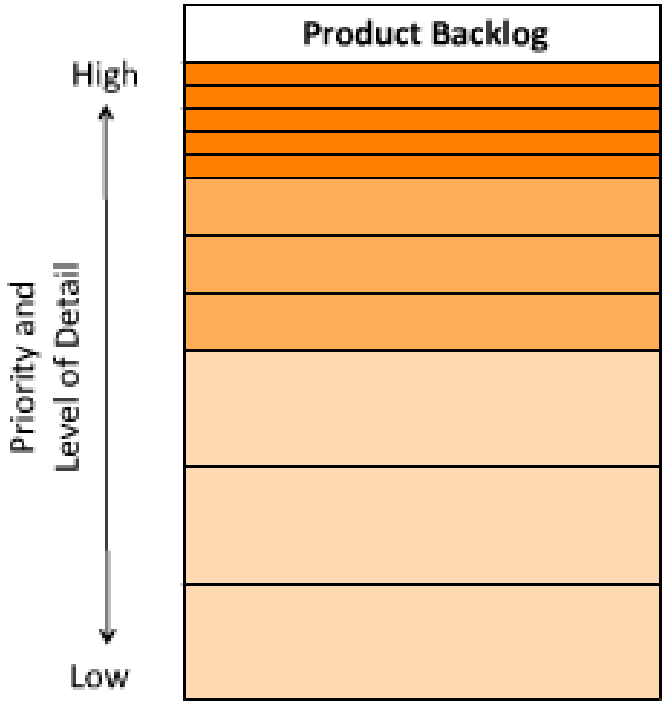
- The bank system allows the account owner to check the current balance and its history till (6 months earlier)

Solution

- The role is obviously, the account owner
- The story can be
  - As an account owner, I can check my balance online to know the current amount and its history
  - Or:
    - As an account owner, I can check my current balance online
    - As an account owner, I can check the history of my balance online

Product backlog

- A list of prioritized user stories form a Product Backlog
- Product backlog is Similar to a list of prioritized "to dos"



http://www.slideshare.net/xpdaysgermany/agile-requirements-engineering-with-scrum-6049200

Product backlog is a live artifact

http://www.slideshare.net/xpdaysgermany/agile-requirements-engineering-with-scrum-6049200

Next Time: User Stories Workshop

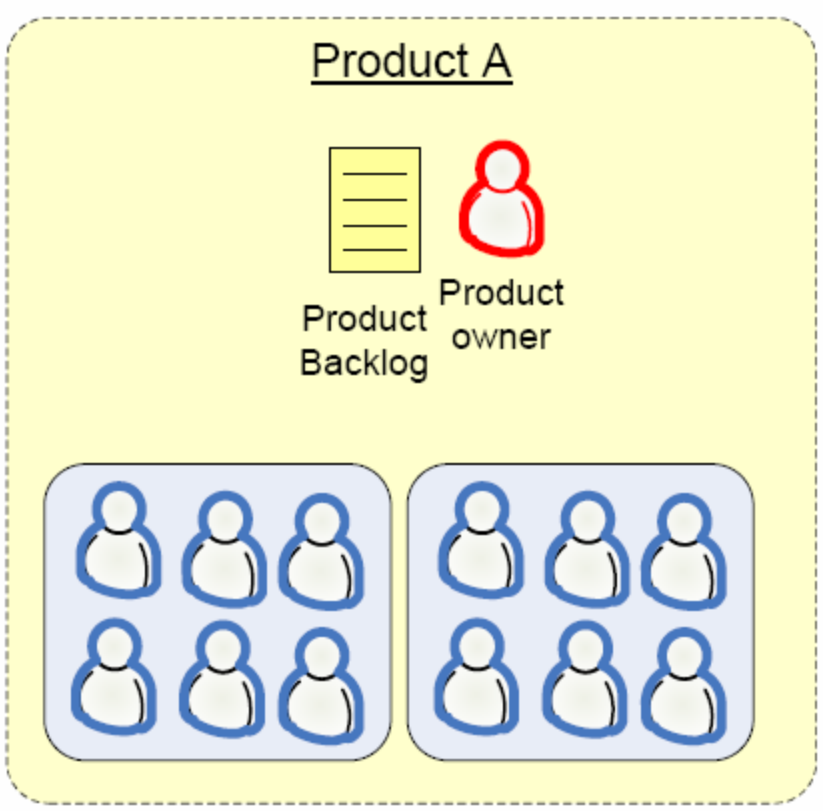
Extracting Formal User stories

What I will give you:

- You will have a description of the project requirements
- The description resembles informal user stories.
- They are divided into components (themes)
- However, all of them are somehow related (it is on system !!)
- Scrum master will share a template of the product backlog with you

Product backlog

- Google Excel file format:
  - Id
  - User story
  - Belong to which Component
  - How to Demo: Success and failure scenarios
  - Depends on which component
  - Priority
  - Story points: relative estimation points
  - Notes

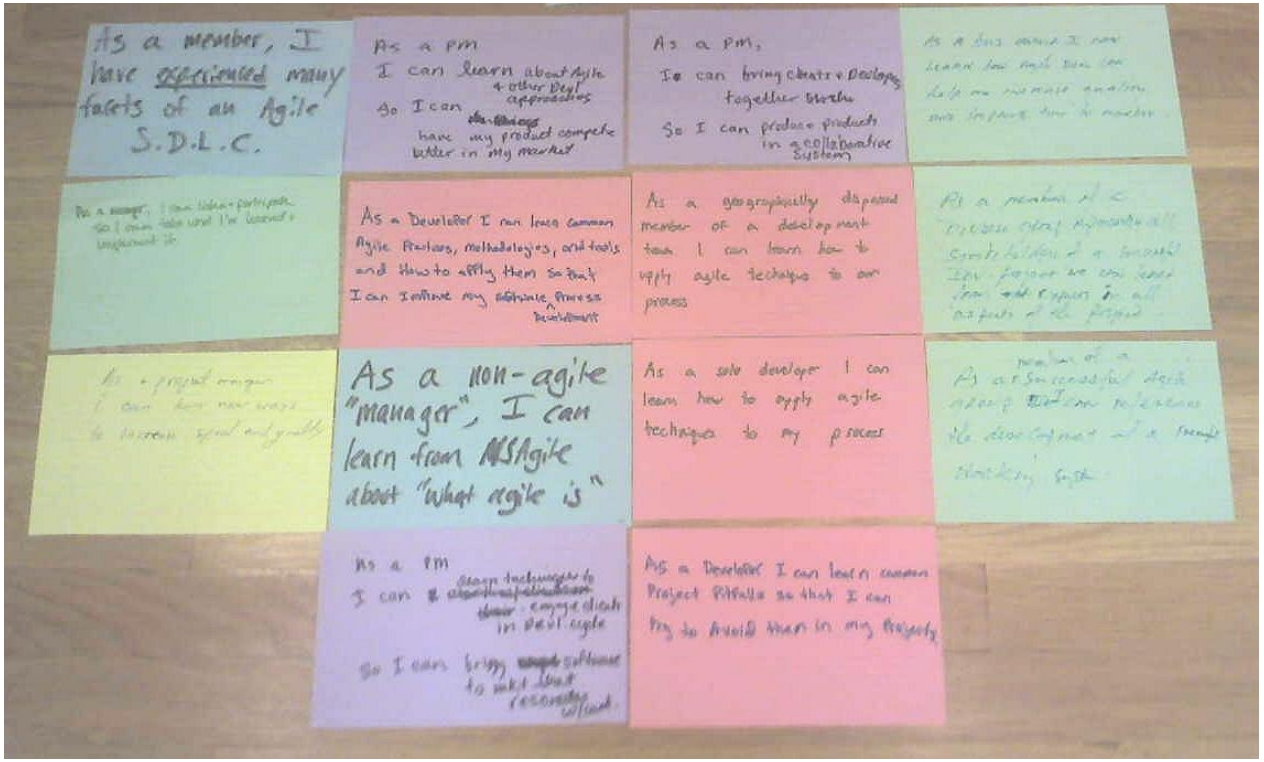


http://www.crisp.se/henrik.kniberg/ScrumAndXpFromTheTrenches.pdf

Snapshot

PRODUCT BACKLOG

ID	STORY / FEATURE /Goal	Dependency	Story Points	*** Priority	Success Demos	Failure Demos	*** Status	*** Spri	Notes
Component 1: Name									
Component 2: Name									


As a member, I have experienced many facets of an Agile S.D.L.C.
As a PM I can learn about Agile & other Dev' approaches So I can have my product compete better in my market
As a PM, I can bring clients + Developers together so that So I can produce products in a collaborative system
As a Developer I can learn common Agile Practices, methodologies, and tools and how to apply them so that I can continue my continuous improvement
As a geographically dispersed member of a development team I can learn how to apply agile techniques to our process
As a non-agile "manager", I can learn from MSAgile about "what agile is"
As a solo developer I can learn how to apply agile techniques to my process
As a PM I can learn techniques to engage clients in the agile So I can bring software to market more accurately
As a Developer I can learn common Project Pitfalls so that I can try to Avoid them in my Projects

What you will do

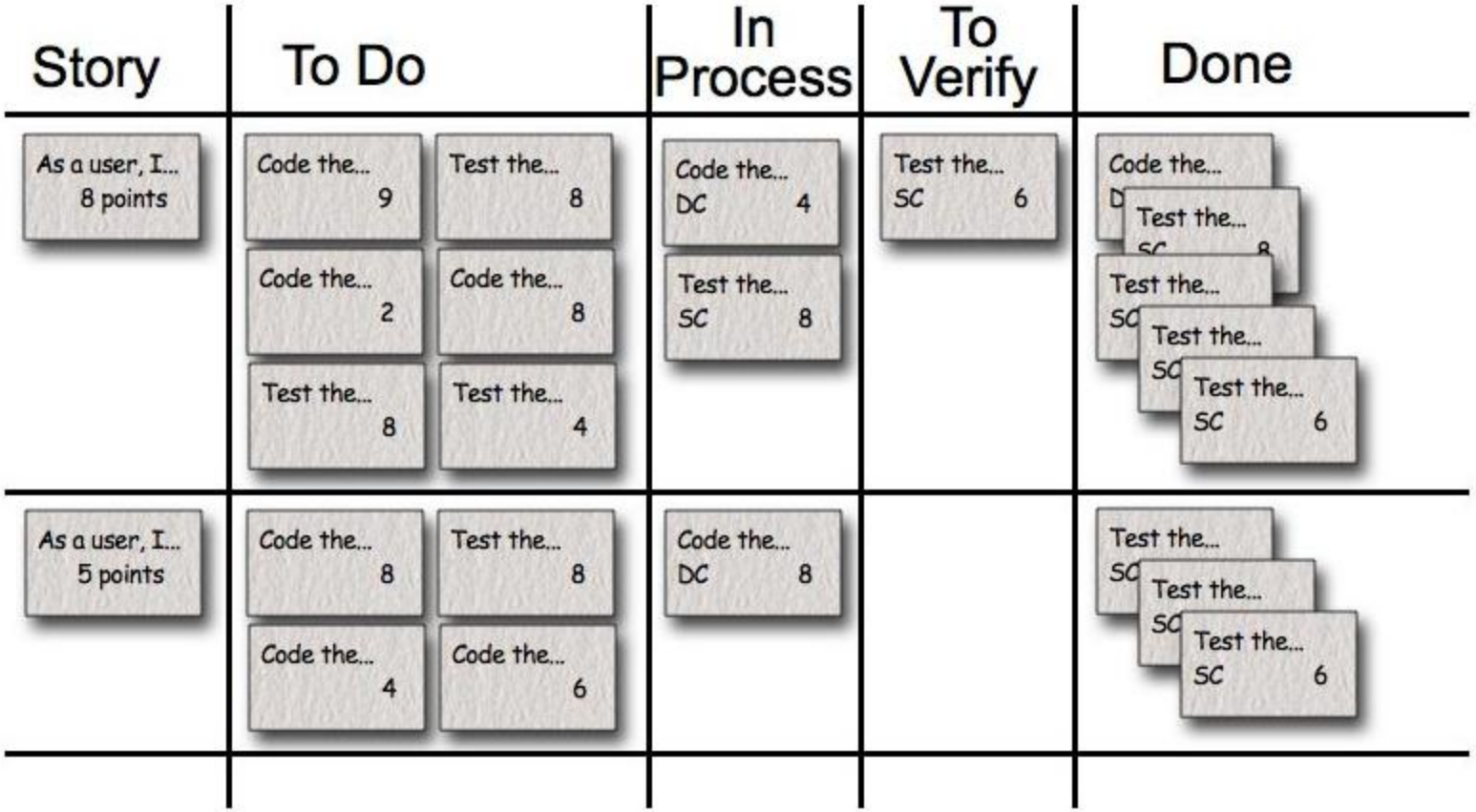
- Brain storming on the different roles in the system (All Company together) (15 minutes)
- Different Components should sit together and produce cards (20 minutes)
- The rest of the time, the cards are discussed with all the company members commenting (even if it is not in their components) (another meeting should be scheduled)
- Advice: One team representative should have the excel file open and start writing in the product backlog the agreed on stories right away.
- Friday midnight: Product backlog submission deadline

How to prepare

- Read the informal requirements document
- Bring index cards
- Come with drafts of user stories from your component

- Optional:
- User Stories:
  - http://www.mountaingoatsoftware.com/system/presentation/file/97/Cohn_SDWest2009_EUS.pdf

Eventually: Component Task board



http://www.mountaingoatsoftware.com/system/hidden_asset/file/29/MockedTaskBoard.jpg