



Requirements

CSCI2340: (Graduate) Software Engineering

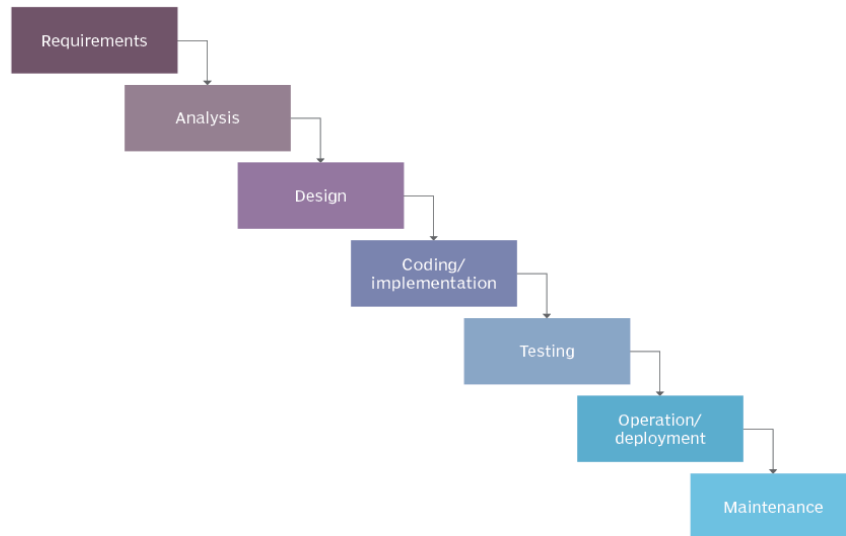
Steven P. Reiss



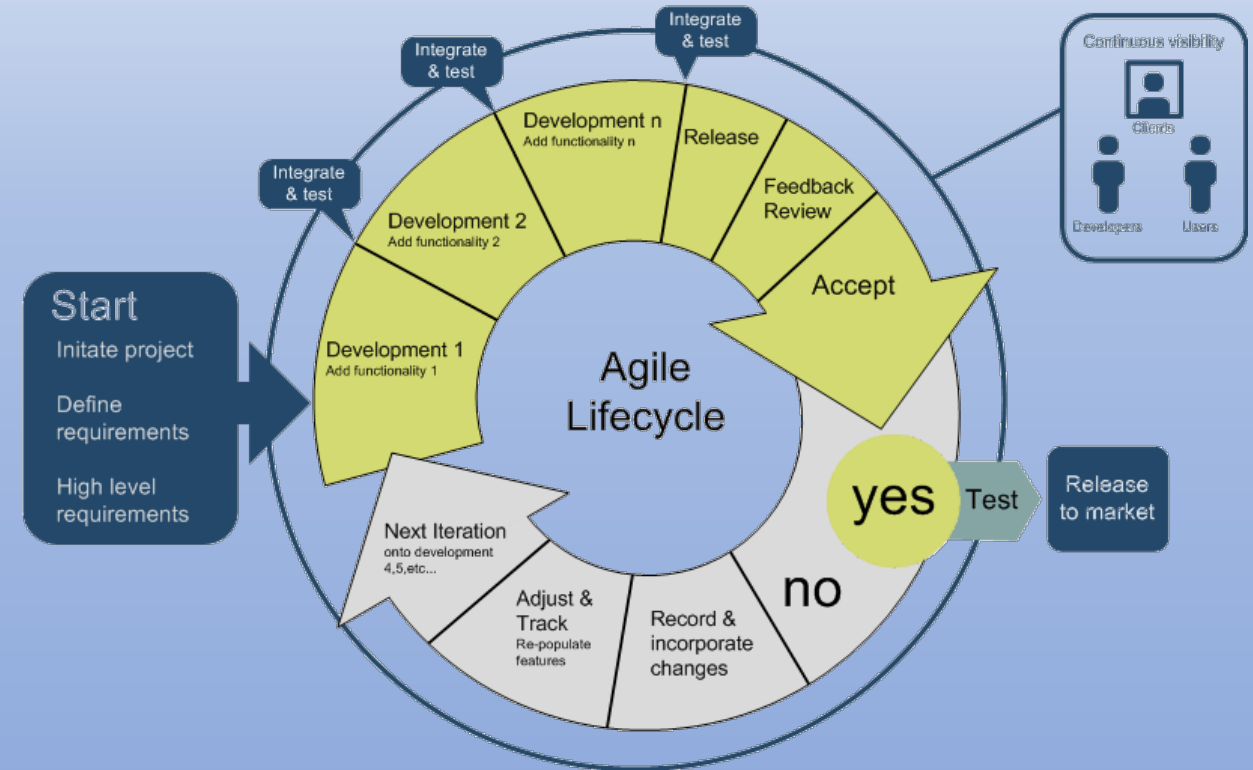
ANY QUESTIONS???

Requirements and the Life-Cycle


Waterfall model



©2019 TECHTARGET. ALL RIGHTS RESERVED. TechTarget



What Are Requirements

- The process of establishing the needs of stakeholders to be solved by the software
 - Requirements are a definition of the PROBLEM
 - What users (stakeholders) want and need
 - Not a definition of the system
 - Or a definition of what will be built
 - Or a definition of how the system is built
 - Or a definition of what technologies to use
 - These come later
- 



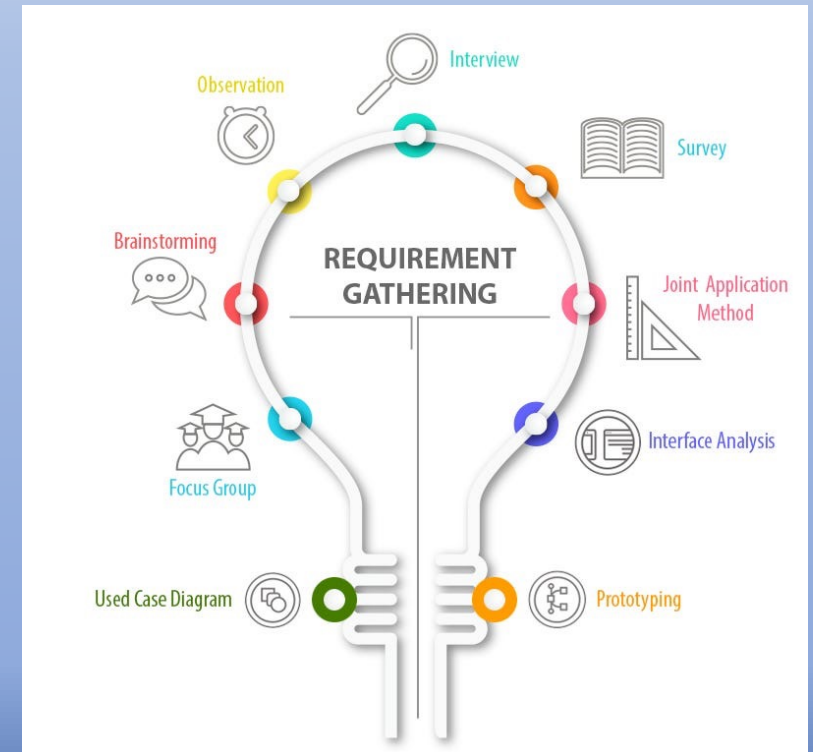
Stakeholders

- Anyone who will care about the eventual product
 - Those who will be using the system
 - Managers
 - Those paying for the software
 - Those interacting with users
- Consider a point-of-sale application
 - Users: check out personal
 - Others: customers, managers, company management



What are Requirements

- Requirements define the system from the user's perspective
 - Specify the problem
 - Specify what is needed in the solution
 - Do not define the implementation
- Requirements constrain the system
 - Constrain specification & design & code
- Requirements provide motivation
 - For specifications and design decisions
- Requirements require users



Importance of Requirements

- Many software projects fail
 - FAFSA; healthcare.gov; ...
- Poor requirements cause over half of these failures
 - 13% fail due to incomplete requirements
 - 12% fail due to lack of user involvement
 - 10% fail due to unrealistic expectations
 - 9% fail due to changing requirements
 - 7% fail because the system is no longer needed



Requirements in the Practice



- Initial Requirements

- Imagine a complete system that would do everything needed
- Include all the bells and whistles
- Include all far-reaching goals and ideas
- Set priorities

- Practical Requirements

- Then narrow this down to a practical initial product
- With the important features emphasized

- Keep initial overall requirements in mind

- Throughout specification, design, coding, maintenance

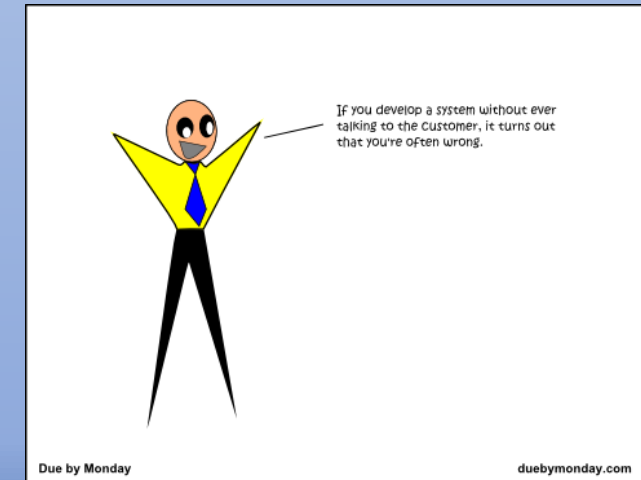
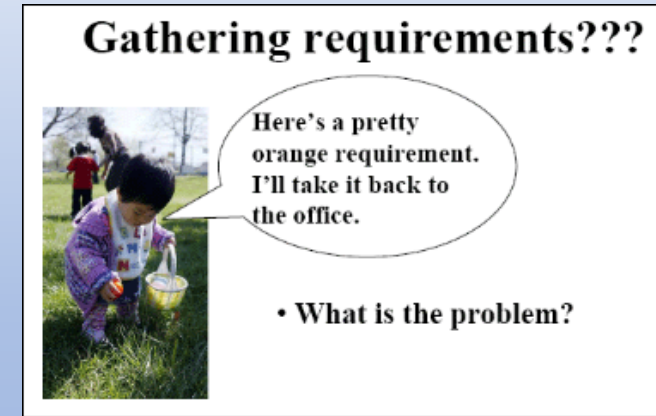
Requirements Process

- **Requirements Analysis is an on-going process**
 - You need a good first approximation
 - Think of it as a dialog with the stakeholders
 - Understand what they want
 - Their understanding what you can do
- **Things are going to change**
 - People change their minds
 - Markets change
 - New opportunities arise



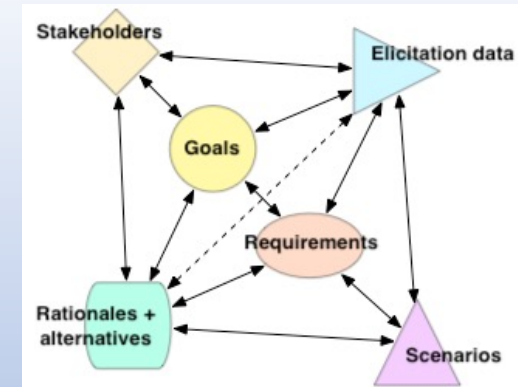
Requirements are Difficult to Obtain

- Talking to users is problematic
 - Communications difficulties
 - Cultural differences, different vocabularies
 - Can't anticipate all questions or issues
 - You won't understand the user's needs and desires
 - Your biases will get in the way
 - Designing & building the system creates opportunities
 - Users are fickle (don't know what they really want)
- Users don't understand what is possible/difficult
- Users are NECESSARY
 - Without them you **will** build the wrong system

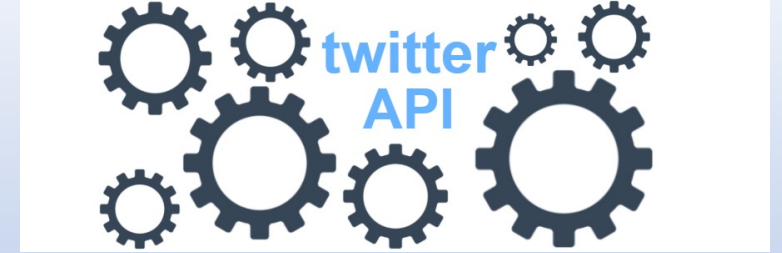


Requirements Goals

- Define the **problem** from the **user's** point of view
- Assess the need for a solution
- Determine the outlines of the ideal solution
- Determine what is required and what is optional
- Determine what is feasible (resource limitations)
- Determine acceptance criteria
- **DO NOT WORRY ABOUT IMPLEMENTATION**



Twitter Data Access



- A data science course wants to have easy access to relevant data. We have a collecting geolocated tweets for a while. Suppose a student in the course wants to analyze how and when the flu becomes prevalent. They should be able to look for tweets containing 'flu' (or similar) over some time period and then view the number of occurrences by state during that time. We want a web app that would let students query, view, and process that data.

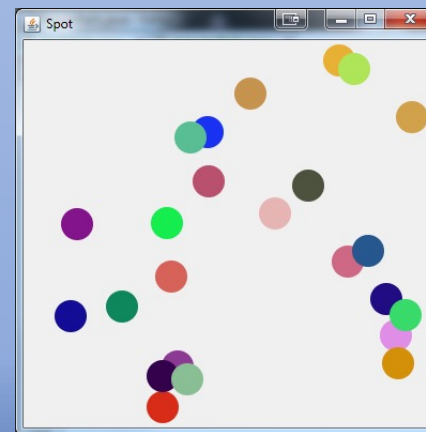
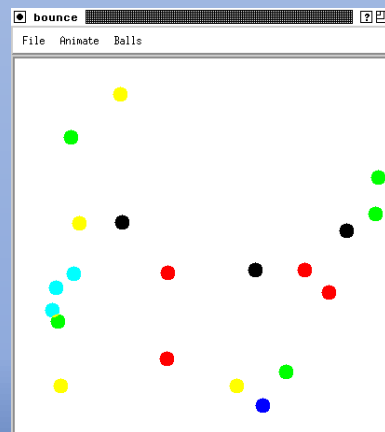
SHORE

I have a HO railroad set up that I put together 25-30 years ago. I want to control this setup from my computer. Moreover, I want to have a high-level control so I can say things like "Take train A out of the yard, go around the outer loop 3 times, then around the inner loop twice, then reverse directions and go around the outer loop twice more before bringing the train back into the yard." At the same time, I want to ensure that the trains do not crash or derail.



Programming Assignment

I like to be amused while I am waiting for my programs to compile or to run. What I would like you to write is a program that displays a set of balls that bounce around in a window in an entertaining way that the user can control.



In-Class EXERCISE

- Task

I like to be amused while I am waiting for my programs to compile or to run. What I would like you to write is a program that displays a set of balls that bounce around in a window in an entertaining way that the user can control.

- Develop requirements for this

- This is a concept, not requirements
- Questions you might want to consider:
 - What do “ball”, “bounce”, “entertaining”, “control” mean
 - What type of interaction is appropriate
 - What might happen in the future
 - What might you want it to look like

- We will split into groups (rows?)

- Introduce yourselves
- Decide who will be “users” and who will be “developers”
- Develop at least 5 basic requirements
- You will have ~5 minutes

Exercise Summary

- What requirements did you produce?

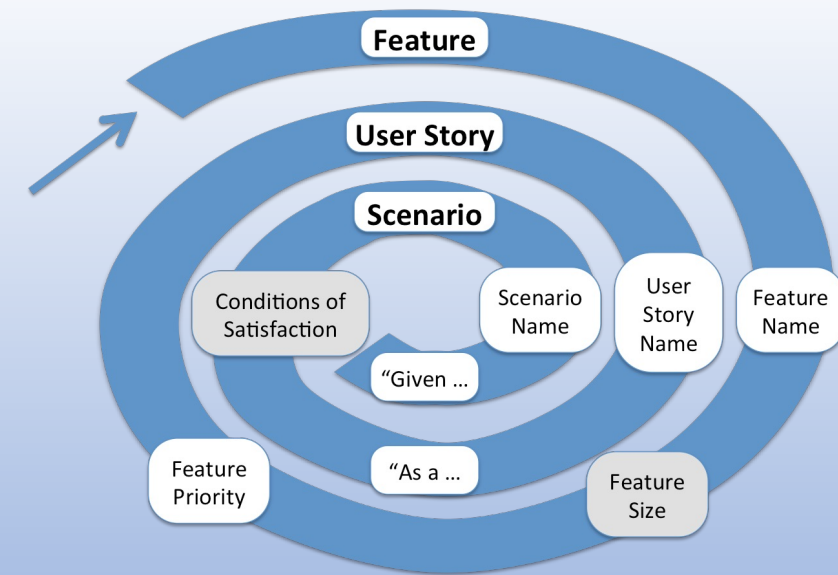
Representing Requirements

- Typically done using Formalized Natural Language
 - Organized as an outline (hierarchical, numbered)
 - With detailed descriptions of what is needed
 - With cross-referencing as appropriate
- Good for following requirements through development
 - Requirements traceability
 - Why is this feature or code or specification or design here
 - What is the importance of this feature
 - How do requirements relate to one another
- Good for formal definition of what is needed
 - Government contracts
- Difficult for the user
 - Can be repetitive, hard to understand
 - Hard to get the overall sense of the proposed application
 - Difficult to ensure completeness, correctness, consistency

Business Requirements Document		
Feature	Definition	Requirement Shopping
Standard based and interoperable messaging protocol	Messaging protocol must be based on industry standards to enable interoperability	
Send Only	Also called Push MEP is simple one-way messaging where a message is sent with no expectation response.	
Receive only	Also called Pull MEP is a message pattern where a non-addressable sender supports the ability to explicitly obtain messages from another application. This can be used for exchanges	
Request/Response exchange	Message pattern consists of one or more request/response pairs. The correlation between request and a response is well defined. In this response maybe deferred and the requesting application may or may not block application processing until a response is received	
Diagnostics	Authentication, diagnostic, logging & routing information should be included in the message and not the payload	
Reliability	Protocol capability to support assured and single delivery to the receiving application with no loss	

Agile Requirements

- Agile Requirements are done as use cases
 - Stories or scenarios
 - Descriptions of common ways the system will be used
 - A story describing a common use of the system
 - From start to end
 - In addition to some standard requirements
- Agile Requirements are done incrementally
 - Basis for sprints, features



Use Cases / Stories / Scenarios

- Detailed description of task
 - Use actual names
 - Be specific
 - Include motivation
 - Be complete
- It is a story
 - Think of a novel
 - Write it as such

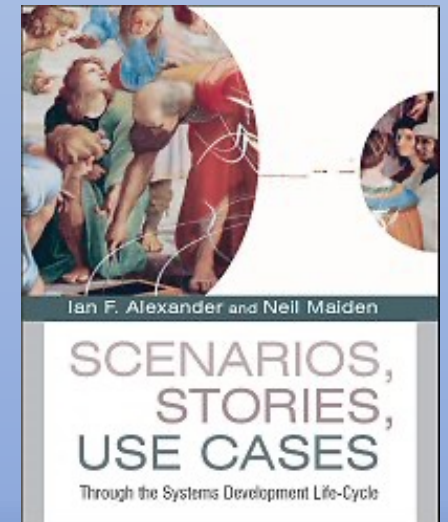


User Story

Professor Reiss needs to compile Code Bubbles. This takes 2-3 minutes. This is too short a time to get work done and too long to do nothing. While he compiles, he wants to move the mouse into a distraction window that shows a set of balls moving around in an interesting manner. To be interesting the balls can eat each other (and grow) and exhibit gravity. He can control the balls to achieve a particular goal, say ending with only the blue ball alive or achieving a stable solution with 2 balls orbiting each other. The program starts when he moves the mouse in and suspends when the mouse is out. The goal should be achieved before the compilation is finished.

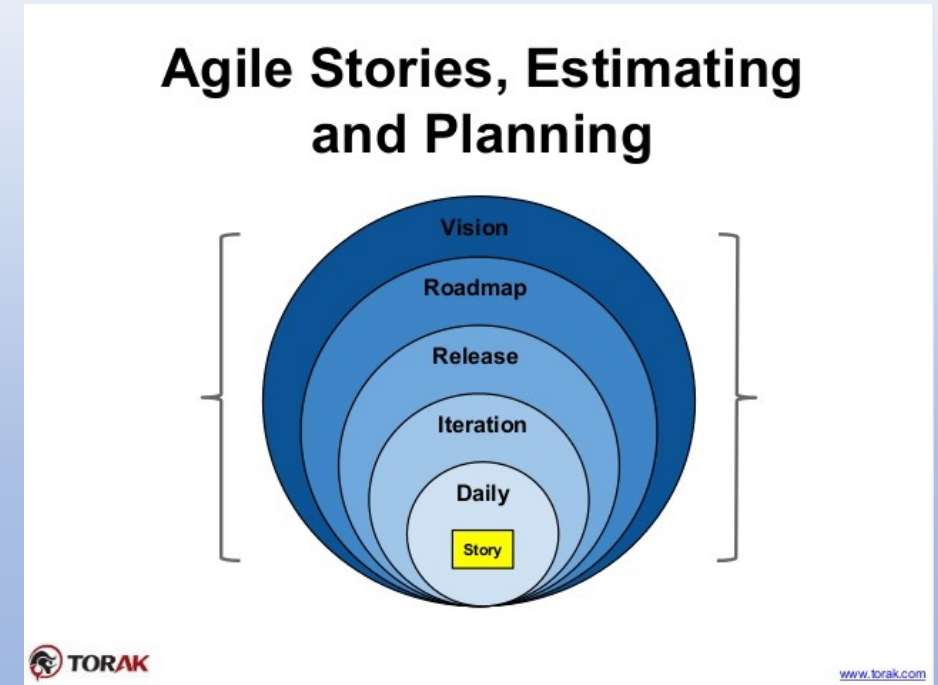
Stories and Scenarios

- **Form a basis for getting user feedback**
 - Give you a sense of what the user wants to do
 - Give the user a sense of what your app can/will do
 - Give the user a context for providing input
 - Give the user a chance to change the story
 - What happens if ...
- **Let the developer give the user feedback**
 - What is too complex
 - What is vague or not well understood
- **Are a basis for specifications**
 - Defining what the system will do



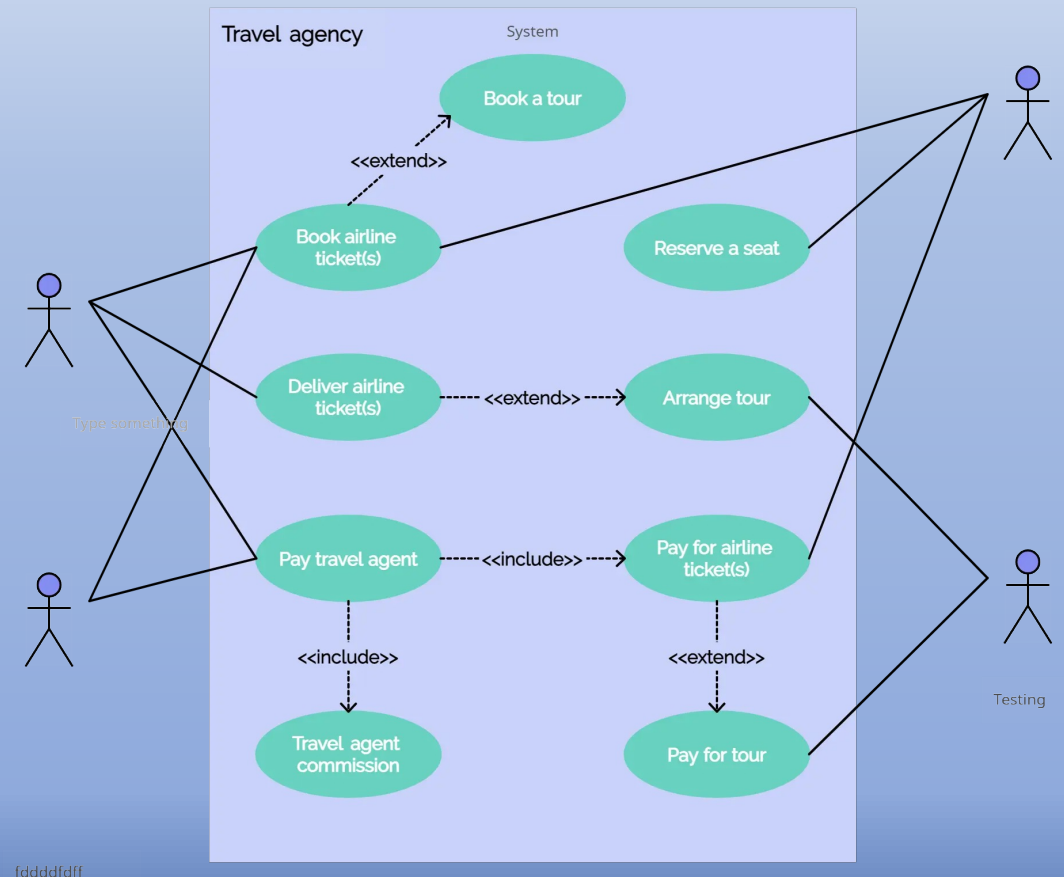
Stories and Scenarios

- Create a set of stories
- Cover the principal uses
- Cover a diverse set of tasks
 - With a diverse set of users
- Should give you a sense of
 - What you will need to do
 - How the application will be used
 - What problem is being solved



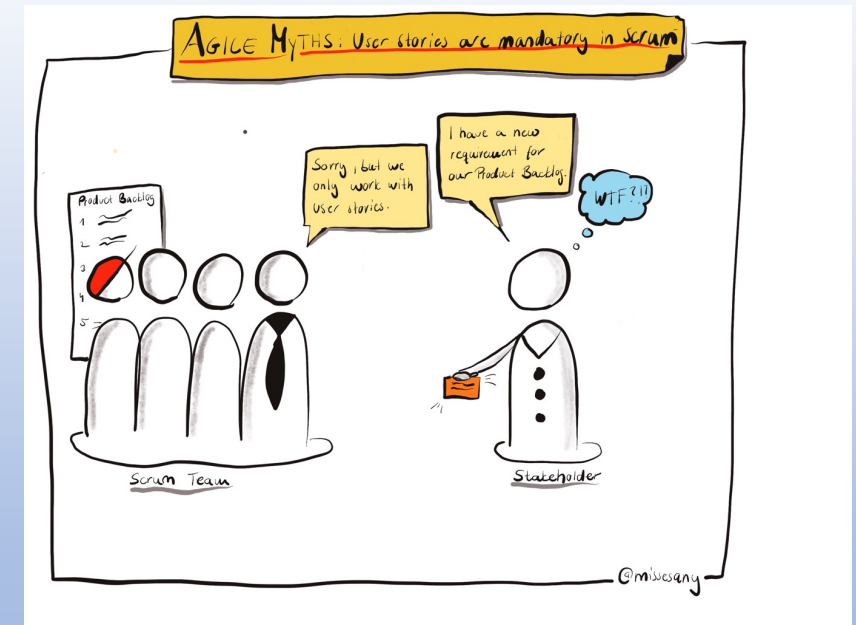
Representing Use Cases

- Can use plain English
 - Possibly with pictures
- UML notation
 - Use-Case Diagrams
 - Actors (people or other systems)
 - Use cases (functionality)
 - Communication links
 - Direct, directed or 2-way
 - Include/exclude other use cases
 - System boundaries



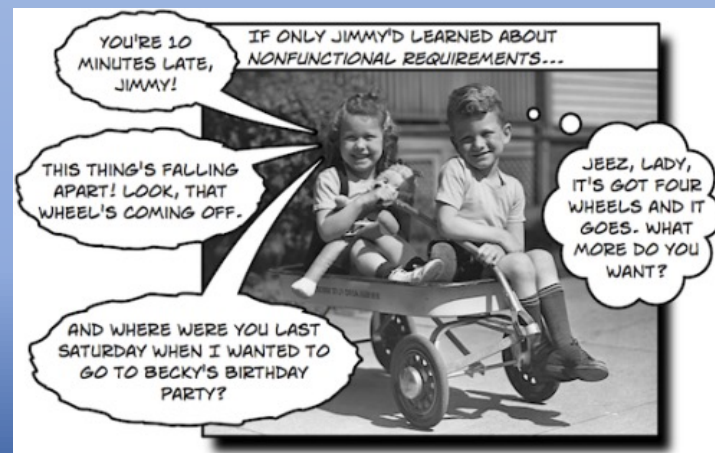
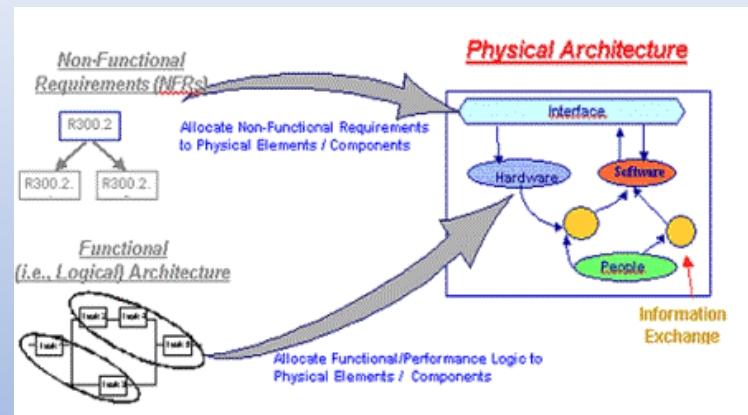
Agile Requirements

- Are easier for the user
 - Just tell a story
- Are more difficult for the developer
 - Harder to achieve requirements traceability
 - Not organized in a meaningful manner
 - Don't really constrain design or implementation
- Are easily made inconsistent
 - Different stories can easily point in opposite directions
- Use a combination of outline requirements & stories



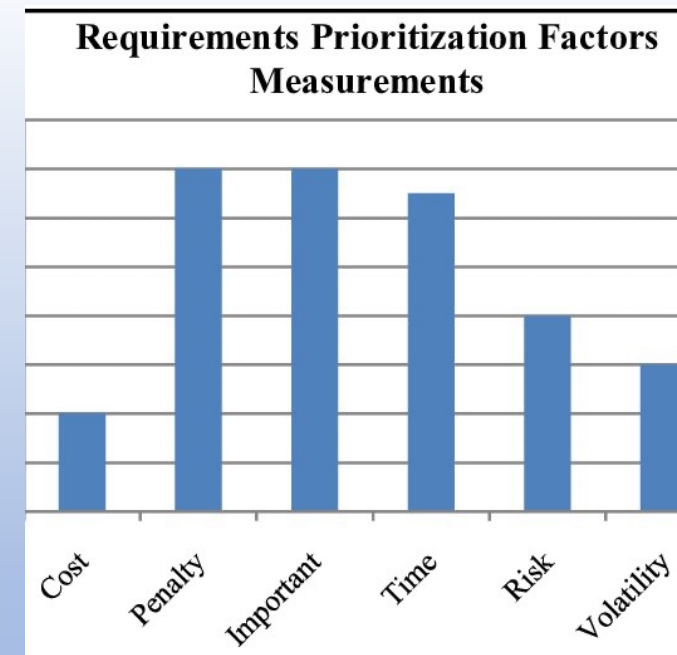
Non-Functional Requirements

- Security constraints
- Privacy constraints
- Anticipated average and maximum loads
 - What should happen if these are exceeded
- Universal Access: Internationalization and accessibility
- Legal issues (money, privacy, children, health, accessibility ...)
- Reliability (up time, data safety)
- Performance
- Time and cost limits
- Fairness



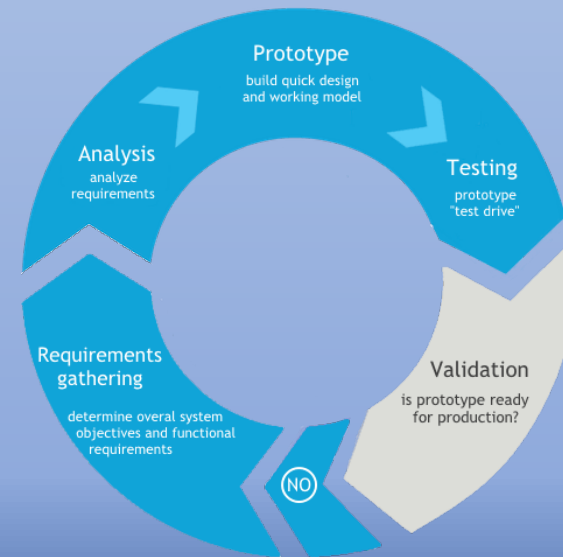
Prioritizing Requirements

- Required vs Optional vs Long-Term
- Minimal viable product
 - MVP
 - What is required to have something that can be used
 - Basis for future development
 - Useful to demonstrate utility of the system
 - Often the basis for start-up development
- Minimal core system
 - What is essential to experiment with requirements
 - What needs to be done before anything else



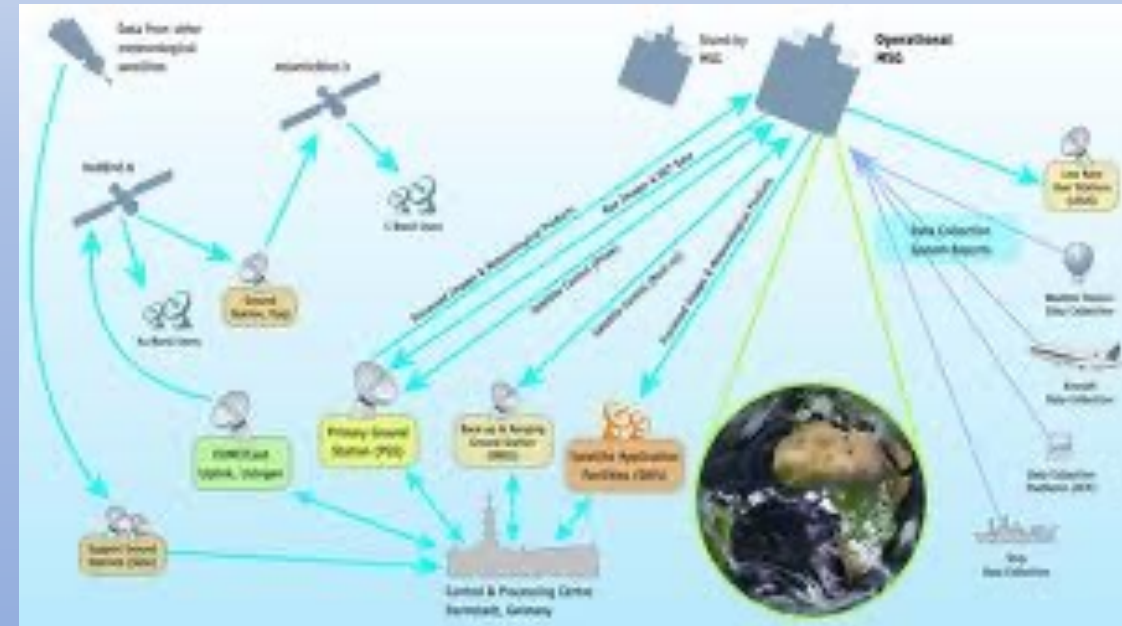
Prototyping

- You might need something concrete to understand
 - What will work, what can work, what will users want, ...
 - Stories give some sense of this, but more might be needed
- You might want to create a prototype
 - Dummy version of the system
 - Can be paper rather than code
 - Just enough to give a sense of possibilities
- Prototypes should be throw-away
 - If you're not willing to throw it away, don't prototype
 - Create a minimal system instead



Getting USER (Stakeholder) Information

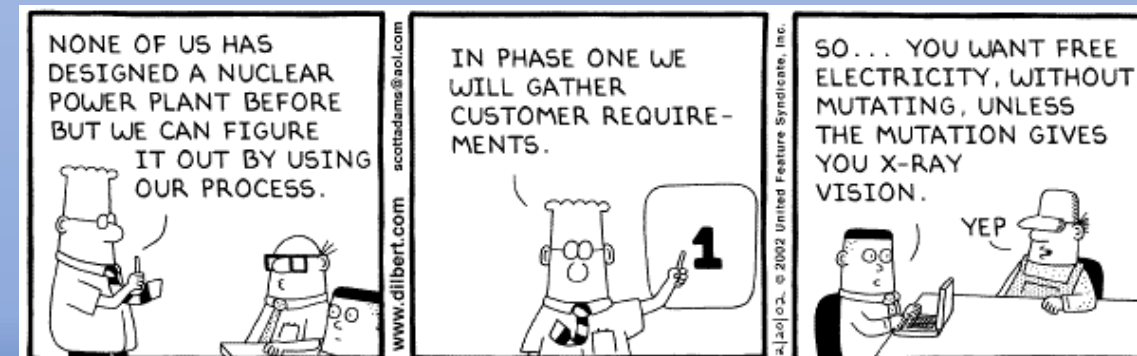
- Talking to users
 - Interviews
 - Focus Groups
 - Questionnaires (Qualtrics, Gforms)
 - Facebook, Mechanical Turk
- Observing users
- Look at competing products



Talking to Users

- Understanding what they want
 - And why they think they want it
- Understanding what is important to them
 - Features, performance, ...
- Vocabulary
 - Don't be afraid to ask questions
- Who is in charge

Dilbert
BY SCOTT ADAMS



Copyright © 2002 United Feature Syndicate, Inc.

Interviewing Potential Users

- Structured versus unstructured interviews
- Interviews require considerable preparation
 - DO YOUR HOMEWORK
 - Determine what information is needed
 - Find out about the interviewees
 - Decide on questions and organization
 - Otherwise, it will be a waste of time
 - Experiment beforehand
- Process
 - Move from general to specific to general (open) questions
 - Summarize the interview
 - In terms of stories, use-cases, scenarios
 - Examples of the use of the system
 - To feed back to the client
 - Follow up



**Users don't
know what they
want.**

Questionnaires and Surveys

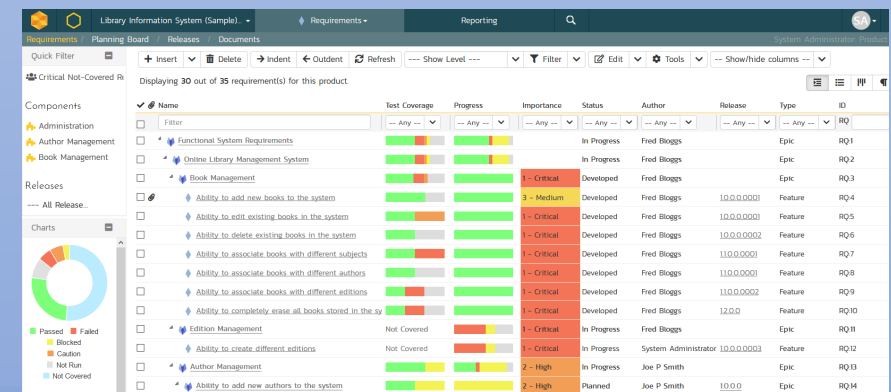
- Who is your target audience (stakeholders)
 - Particular users or the general public
 - Others that may be affected by the system
 - Avoiding bias in audience selection
- Needs a lot of thought
 - What information is needed
 - Phrasing questions so as not to bias the outcomes
 - Determining how to interpret the results
 - Experiment beforehand
- Likert scales
 - Range of 1-7 (or 1-5) where ... rate X
 - Provides a means of compiling results
- End with open questions (won't be answered by everyone)

The image shows a screenshot of a '360 Degree Assessment' form. At the top, it says 'Staff Self-Assessment for [Name of your role]' and '360 Degree Assessment'. Below this, it asks 'Please indicate your level of agreement with the following statements on a scale from 1 to 5 (1 = Strongly Agree, 5 = Strongly Disagree)'. The form is divided into two sections: 'Personal & Technical' and 'Working & Supervising'. Each section contains a list of statements and a scale from 1 to 5. The 'Personal & Technical' section includes statements like 'I am a good team player', 'I am a good listener', 'I am a good communicator', 'I am a good problem solver', 'I am a good decision maker', and 'I am a good leader'. The 'Working & Supervising' section includes statements like 'I am a good manager', 'I am a good supervisor', 'I am a good coach', 'I am a good mentor', 'I am a good role model', and 'I am a good team leader'. The form is a standard questionnaire used for performance evaluation.



Requirements Tools

- Provide a means for organizing requirements
 - Geared toward standard requirements
 - Easy to add, change, etc. requirements
 - Ensure requirements are always available
- JAMA, IBM Doors, CaseComplete



Name	Test Coverage	Progress	Importance	Status	Author	Release	Type	ID
Functional System Requirements	100%	100%	1 - Critical	In Progress	Fred Bloggs		Epic	RQ1
Online Library Management System	100%	100%	1 - Critical	In Progress	Fred Bloggs		Epic	RQ2
Book Management	100%	100%	1 - Critical	Developed	Fred Bloggs		Epic	RQ3
Ability to add new books to the system	100%	100%	3 - Medium	Developed	Fred Bloggs	10.0.0.0001	Feature	RQ4
Ability to edit existing books in the system	100%	100%	1 - Critical	Developed	Fred Bloggs	10.0.0.0002	Feature	RQ5
Ability to delete existing books in the system	100%	100%	1 - Critical	Developed	Fred Bloggs	10.0.0.0003	Feature	RQ6
Ability to associate books with different subjects	100%	100%	1 - Critical	Developed	Fred Bloggs	11.0.0.0001	Feature	RQ7
Ability to associate books with different authors	100%	100%	1 - Critical	Developed	Fred Bloggs	11.0.0.0002	Feature	RQ8
Ability to associate books with different editions	100%	100%	1 - Critical	Developed	Fred Bloggs	11.0.0.0003	Feature	RQ9
Ability to completely erase all books stored in the system	100%	100%	1 - Critical	Developed	Fred Bloggs	12.0.0	Feature	RQ10
Edition Management	Not Covered	100%	1 - Critical	In Progress	Fred Bloggs		Epic	RQ11
Ability to create different editions	Not Covered	100%	1 - Critical	In Progress	System Administrator	10.0.0.0003	Feature	RQ12
Author Management	100%	100%	2 - High	In Progress	Joe P. Smith		Epic	RQ13
Ability to add new authors to the system	100%	100%	2 - High	Planned	Joe P. Smith	10.0.0	Epic	RQ14

Requirements Research

- Using AI (NLP) techniques
 - For obtaining natural-language requirements
 - For analyzing natural-language requirements
- Other means of eliciting requirements
 - App-store based techniques
- Defining fairness
- Cross-domain requirements linking
- Analyzing privacy policy of third-party libraries



PROJECT

- Project teams will be assigned tomorrow
 - You will be notified via e-mail
 - Next class we will have an initial team meeting
 - [Form for providing project preferences](#)
- Current Popular Projects
 - DJ Mix
 - LLAMA
 - Speech

PROJECT Homework

- Once projects are assigned
 - Create a shared requirements document for your project
 - Start adding requirements for your project
 - Start thinking of who to interview or poll or work with
 - Identify and start talking to potential users
 - Add (individually) to the requirements document
- For next class's project meeting
 - Think about your role in the project
 - Think about how you want the project to go

Homework (from last time)

- Download and install Code Bubbles
 - Need to install Eclipse first if not already there
- Run Code Bubbles on a project
 - Existing Eclipse workspace
 - Existing Java project
 - New code (hello world)
- Relate experiences (pro and con) in a canvas hand-in
- Due 9/12 (Thursday)

Homework (Requirements)

- Develop requirements for the programming assignment
 - For what **you** want to build, not in general
 - Use input from our earlier exercise if desired
 - Hand in via canvas. **Due 9/12 (Thursday)**
 - We will use this in the next class
- Requirements Quality Assistant Game
 - <https://www.ibm.com/internet-of-things/learn/RQA-interactive/#/>

Further Reading

- Text Book: Chapter 3
- (Optional) View the IBM Doors preview video
 - <https://www.ibm.com/products/ibm-engineering-requirements-management-doors-next>
- (Optional) View the CaseComplete preview video
 - <https://casecomplete.com/learn>
 - Look at the follow-ups (CaseComplete 101)