Telerik Academy

http://academy.telerik.com

Software Engineering Fundamentals

Software Development Practices and Methodologies

Telerik Software Academy

Learning & Development

http://academy.telerik.com

Telerik Academy

Table of Contents

- Software engineering overview
  - Requirements
  - Design
  - Construction
  - Testing
  - Project management

- Development methodologies overview
  - The waterfall development process
  - Heavyweight methodologies
  - Agile methodologies, SCRUM and XP

Telerik Academy

Software Engineering

Requirements, Design, Construction, Testing

What is Software Engineering?

Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software.

Definition by IEEE

Software Engineering

- Software engineering is:
  - An engineering discipline that provides knowledge, tools, and methods for:
    - Defining software requirements
    - Performing software design
    - Software construction
    - Software testing
    - Software maintenance tasks
    - Software project management

Software Development Activities

- Software development always includes the following activities (to some extent):
  - Requirements analysis
  - Design
  - Construction
  - Testing (sometimes)

} Software Project Management

- These activities do not follow strictly one after another (depends on the methodology)!
  - Often overlap and interact

Telerik Academy

Software Requirements

Functional & Non-functional Requirements, SRS, User Story Cards

Software Requirements

- Software requirements define the functionality of the system
  - Answer the question "what?", not "how?"
  - Define constraints on the system
- Two kinds of requirements
  - Functional requirements
  - Non-functional requirements

Requirements Analysis

- Requirements analysis starts from a vision about the system
  - Customers don't know what they need!
  - Requirements come roughly and are specified and extended iteratively
- The outcome is the Software Requirements Specification (SRS) or set of User Stories
- Prototyping is often used, especially for the user interface (UI)

Software Requirements Specification (SRS)

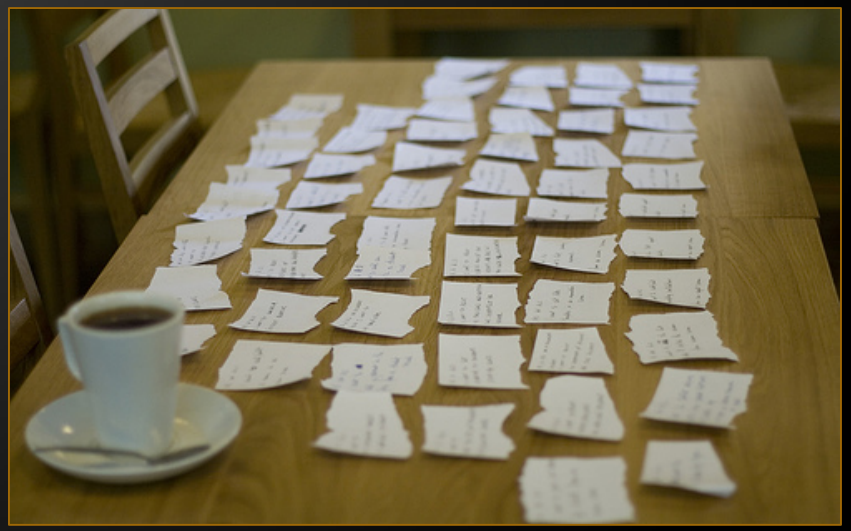
- The Software Requirements Specification (SRS) is a formal requirements document
- It describes in details:
  - Functional requirements
    - Business processes
    - Actors and use-cases
  - Non-functional requirements
    - E.g. performance, scalability, constraints, etc.

Agile Requirements and User Stories

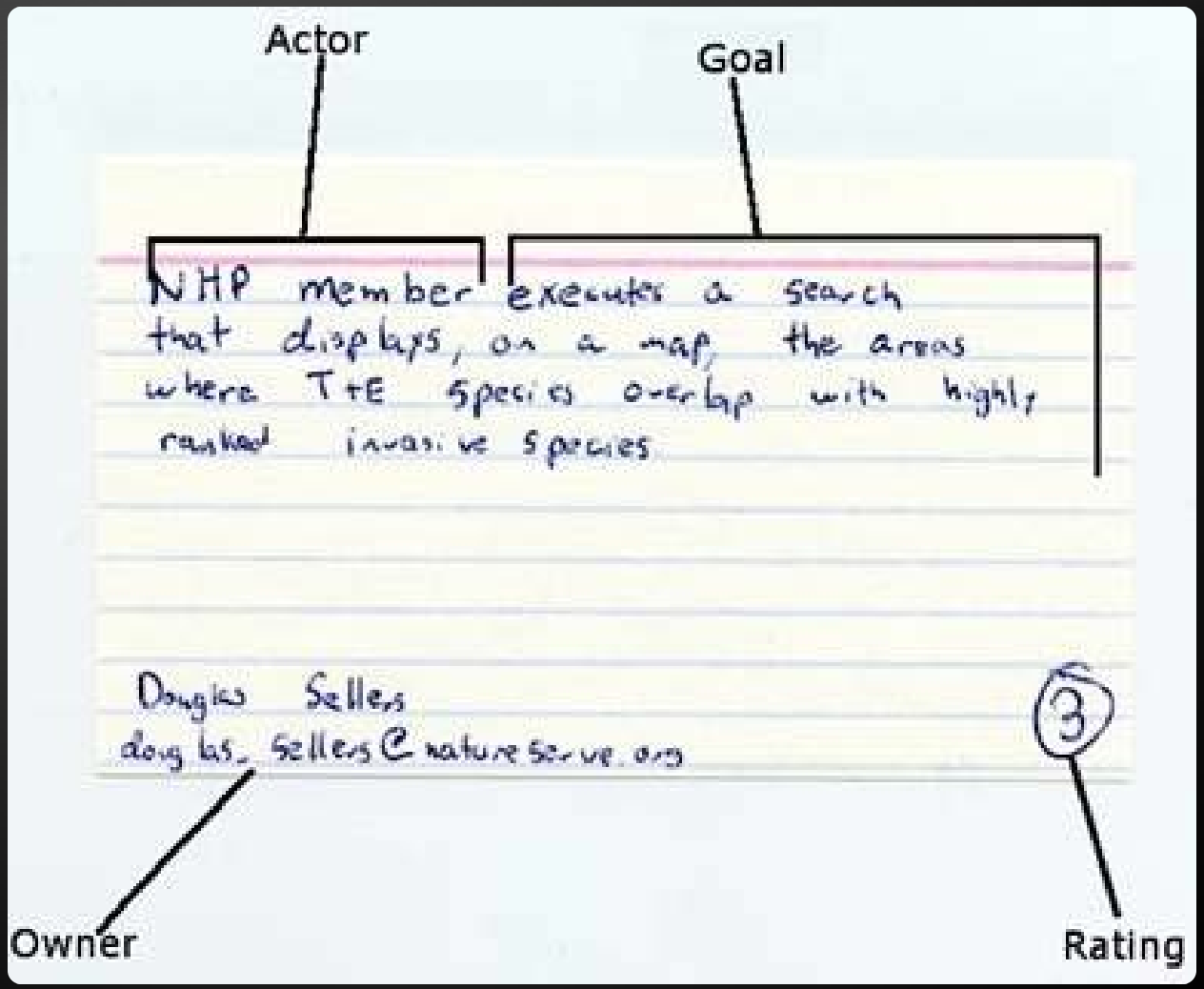
- Requirements specifications are too heavy
  - Does not work well in dynamic projects that change their requirements every day
- Agile development needs agile requirements
  - Split into small iterations
- How to split the requirements?
  - Use simple, informal requirements description
  - User story: a small feature that brings some value to the end-user

What is User Story?

- User story
  - User needs to accomplish something
  - Written informal (in words / images / sketches)
  - Looks like use-case but is different (less formal)
- User stories have
  - Actor (who?)
  - Goal (what?, why?)
  - Other info
    - Owner, estimate, …



User Story – Example



Software Requirements

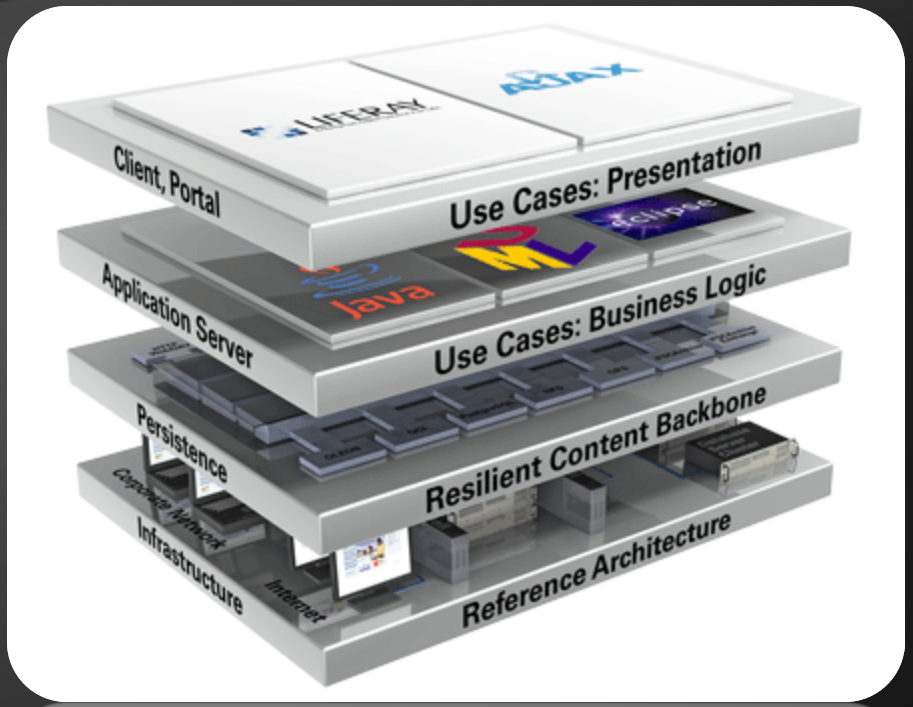
- It is always hard to describe and document the requirements in comprehensive way
  - Good requirements save time and money
- Requirements always change during the project!
  - Good requirements reduces the changes
  - Prototypes significantly reduce changes
  - Agile methodologies are flexible to changes
    - Incremental development in small iterations

Telerik Academy

Software Requirements Specifications (SRS), User Stories and UI Prototypes

Live Demo

Telerik Academy

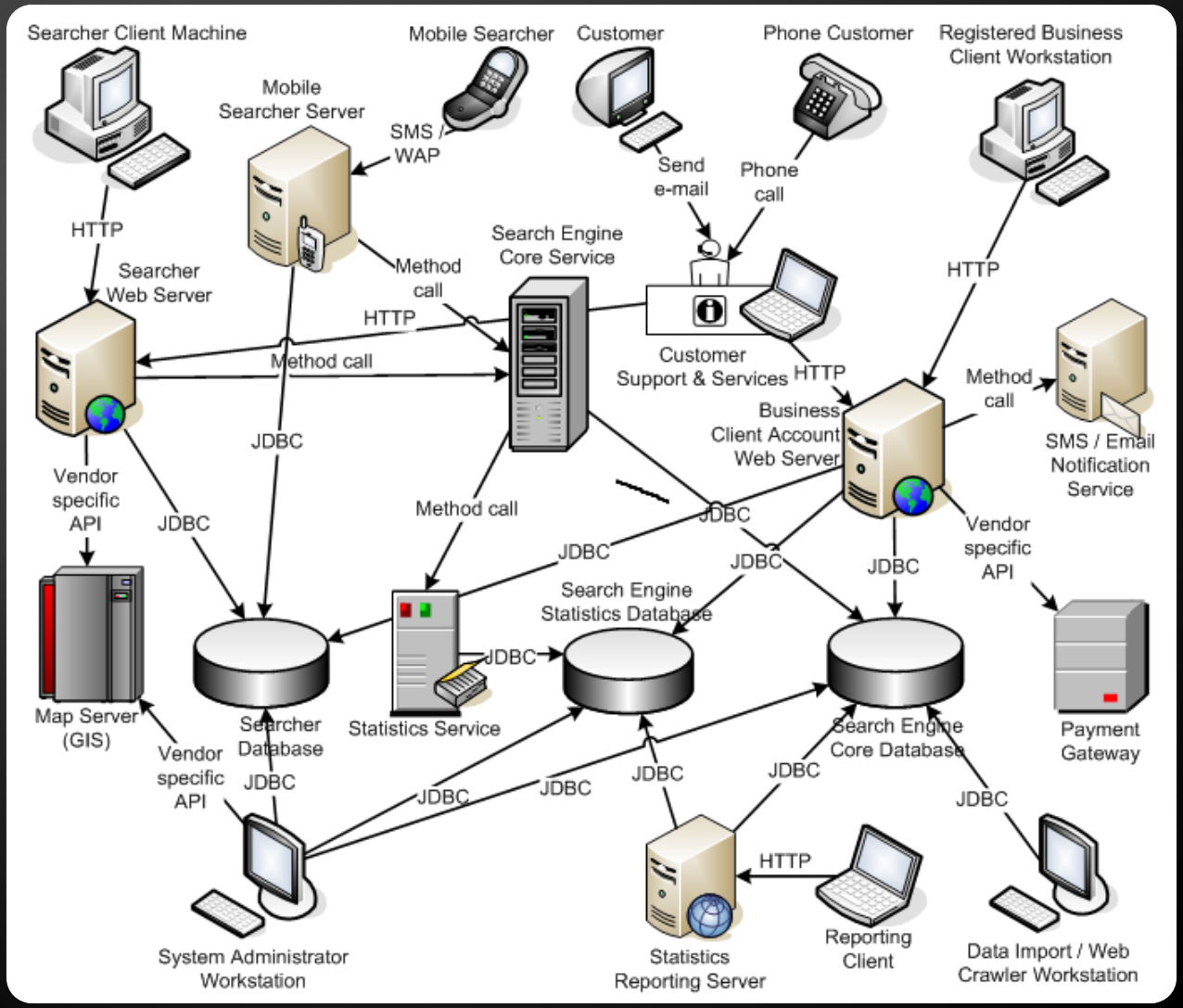


Software Architecture and Software Design

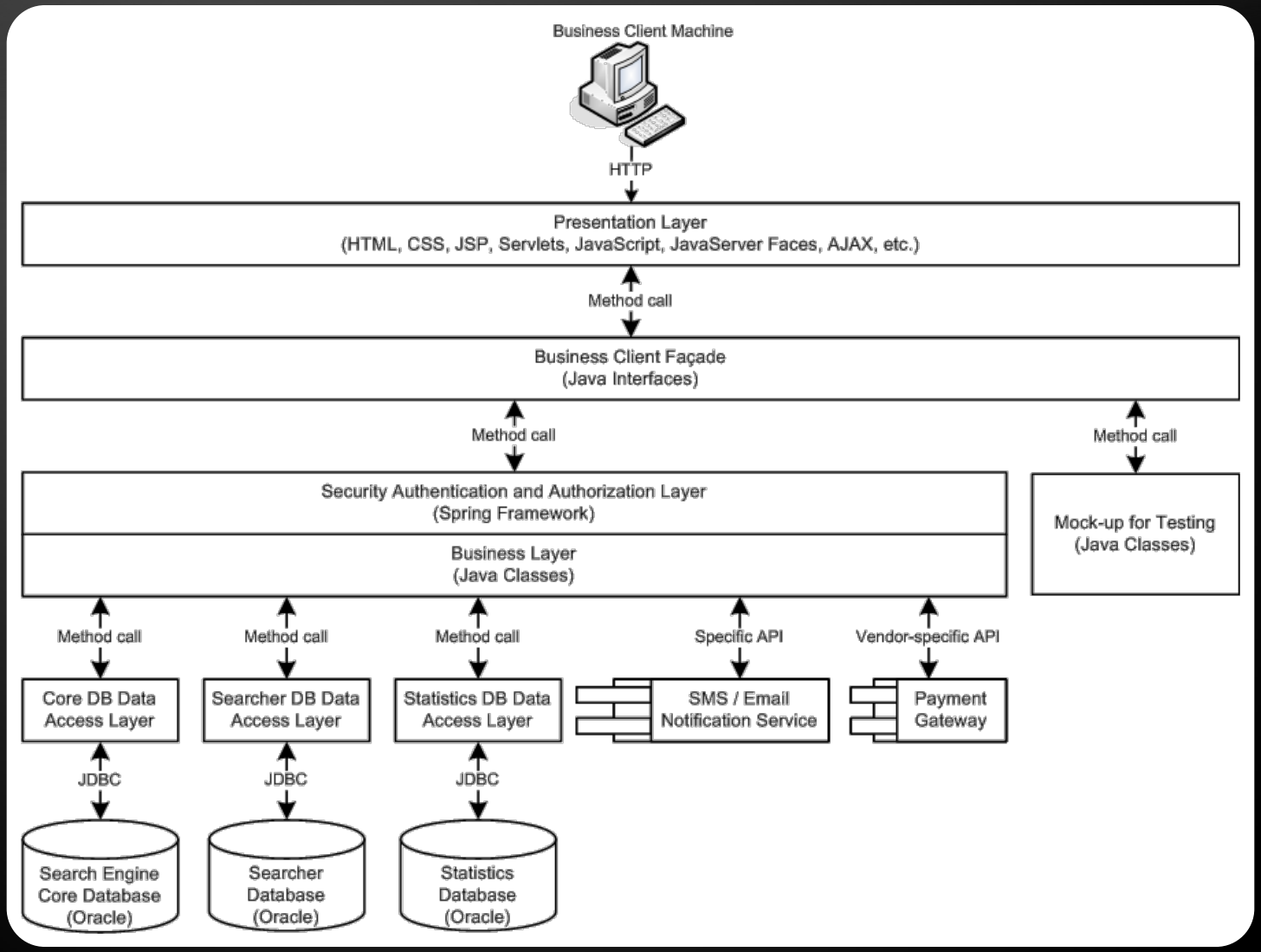
Software Architecture and Software Design

- Software design is a technical description (blueprints) about how the system will implement the requirements
- The system architecture describes:
  - How the system will be decomposed into subsystems (modules)
  - Responsibilities of each module
  - Interaction between the modules
  - Platforms and technologies

System Architecture Diagram – Example



Software Architecture Diagram – Example



Software Design

- Detailed Design
  - Describes the internal module structure
  - Interfaces, data design, process design
- Object-Oriented Design
  - Describes the classes, their responsibilities, relationships, dependencies, and interactions
- Internal Class Design
  - Methods, responsibilities, algorithms and interactions between them

Software Design Document (SDD)

- The Software Design Document (SDD)
  - Formal description of the architecture and design of the system
- It contains:
  - Architectural design
    - Modules and their interaction (diagram)
  - For each module
    - Process design (diagrams)
    - Data design (E/R diagram)
    - Interfaces design (class diagram)

Telerik Academy

Software Design Document

Live Demo

Telerik Academy

Software Construction

Implementation, Unit Testing, Debugging, Integration

Software Construction

- During the software construction phase developers create the software
  - Sometimes called implementation phase
- It includes:
  - Internal method design
  - Writing the source code
  - Writing unit tests (optionally)
  - Testing and debugging
  - Integration

Writing the Code

- Coding is the process of writing the programming code (the source code)
  - The code strictly follows the design
  - Developers perform internal method design as part of coding
- The source code is the output of the software construction process
  - Written by developers
  - Can include unit tests

Testing the Code

- Testing checks whether the developed software conforms to the requirements
  - Aims to identify defects (bugs)
- Developers test the code after writing it
  - At least run it to see the results
  - Unit testing works better
    - Units tests can be repeated many times

- System testing is done by the QA engineers
  - Unit testing is done by developers

Debugging

- Debugging aims to find the source of already identified defect and to fix it
  - Performed by developers
- Steps in debugging:
  - Find the defect in the code
    - Identify the source of the problem
    - Identify the exact place in the code causing it
  - Fix the defect
  - Test to check if the fix is working correctly

Integration

- Integration is putting all pieces together
  - Compile, run and deploy the modules as a single system
  - Test to identify defects
- Integration strategies
  - Big bang, top-down and bottom-up
  - Continuous integration

Coding != Software Engineering

- Inexperienced developers consider coding the core of development
  - In most projects coding is only 20% of the project activities!
  - The important decisions are taken during the requirements analysis and design
  - Documentation, testing, integration, maintenance, etc. are often disparaged
- Software engineering is not just coding!
  - Programmer != software engineer

Telerik Academy

Software Verification and Testing

Software Verification

- What is software verification?
  - It checks whether the developed software conforms to the requirements
  - Performed by the Software Quality Assurance Engineers (QA engineers)
- Two approaches:
  - Formal reviews and inspections
  - Different kinds of testing
- Cannot certify absence of defects!
  - Can only decrease their rates

Software Testing

- Testing checks whether the developed software conforms to the requirements
- Testing aims to find defects (bugs)
  - Black-box and white-box tests
  - Unit tests, integration tests, system tests, acceptance tests
  - Stress tests, load tests, regression tests
  - Tester engineers can use automated test tools to record and execute tests

Software Testing Process

- Test planning
  - Establish test strategy and test plan
  - During requirements and design phases
- Test development
  - Test procedures, test scenarios, test cases, test scripts
- Test execution
- Test reporting
- Retesting the defects

Test Plan and Test Cases

- The test plan is a formal document that describes how tests will be performed
  - List of test activities to be performed to ensure meeting the requirements
  - Features to be tested, testing approach, schedule, acceptance criteria
- Test scenarios and test cases
  - Test scenarios – stories to be tested
  - Test cases – tests of single function

Telerik Academy

Test Plans and Test Cases

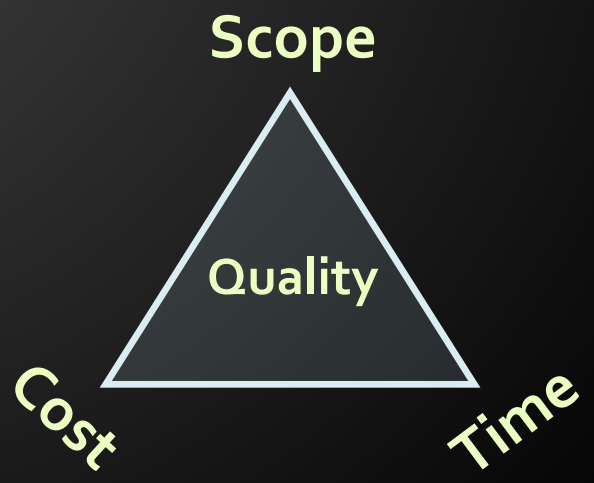
Live Demo

Telerik Academy

Software Project Management

What is Project Management?

- Project management is the discipline of organizing and managing work and resources in order to successfully complete a project
- Successfully means within defined scope, quality, time and cost constraints
- Project constraints:



What is Software Project Management?

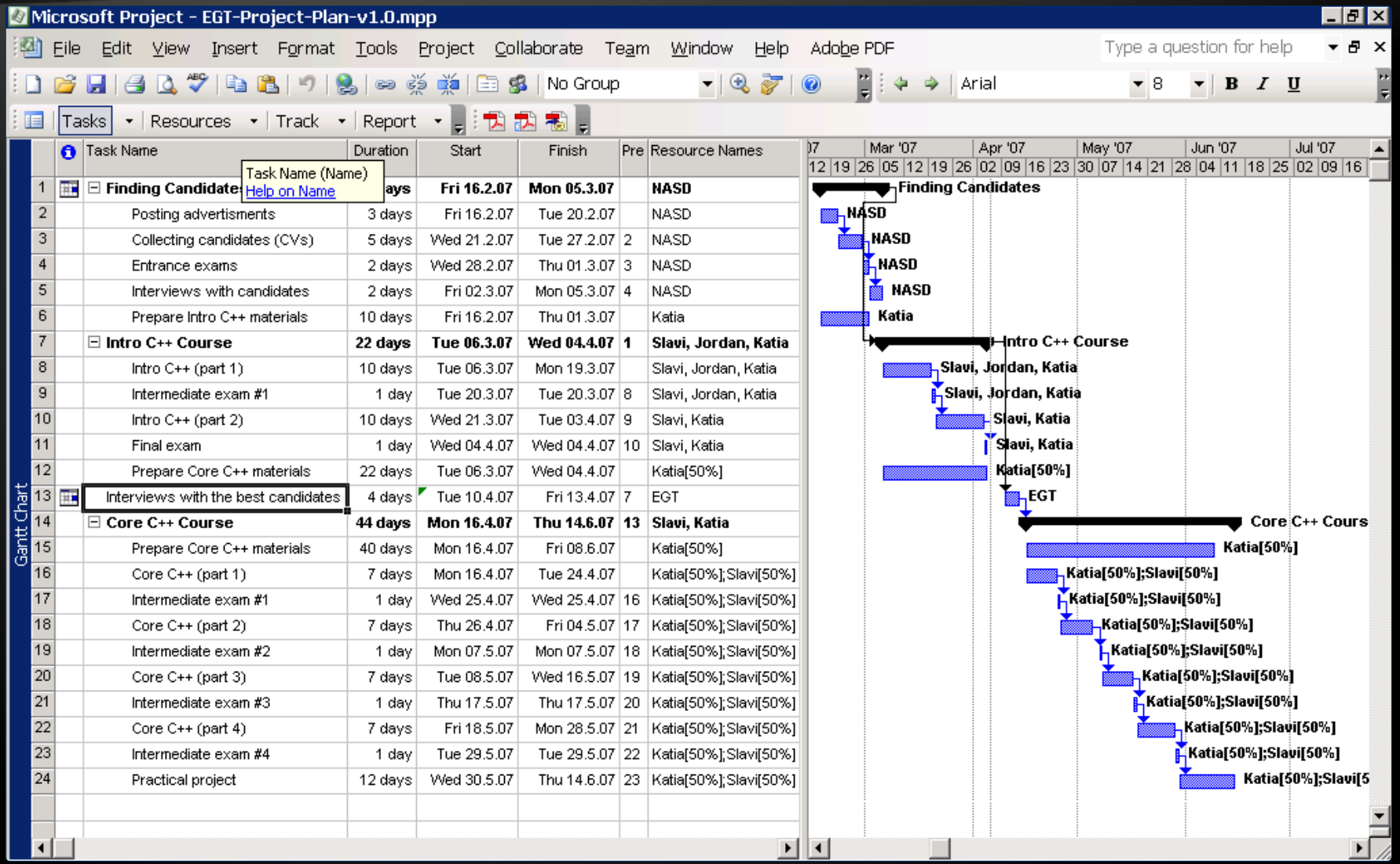
- Software project management
  - Management discipline about planning, monitoring and controlling software projects
- Project planning
  - Identify the scope, estimate the work involved, and create a project schedule
- Project monitoring and control
  - Keep the team up to date on the project's progress and handle problems

What is Project Plan?

- The project plan is a document that describes how the work on the project will be organized
  - Contains tasks, resources, schedule, milestones, etc.
  - Tasks have start, end, assigned resources (team members), % complete, dependencies, nested tasks, cost, etc.
- Project management tools simplify creating and monitoring project plans

Telerik Academy

Project Plan – Example



	Task Name	Duration	Start	Finish	Pre	Resource Names
1	**Finding Candidates**	**days**	**Fri 16.2.07**	**Mon 05.3.07**		**NASD**
2	Posting advertisments	3 days	Fri 16.2.07	Tue 20.2.07		NASD
3	Collecting candidates (CVs)	5 days	Wed 21.2.07	Tue 27.2.07	2	NASD
4	Entrance exams	2 days	Wed 28.2.07	Thu 01.3.07	3	NASD
5	Interviews with candidates	2 days	Fri 02.3.07	Mon 05.3.07	4	NASD
6	Prepare Intro C++ materials	10 days	Fri 16.2.07	Thu 01.3.07		Katia
7	**Intro C++ Course**	**22 days**	**Tue 06.3.07**	**Wed 04.4.07**	**1**	**Slavi, Jordan, Katia**
8	Intro C++ (part 1)	10 days	Tue 06.3.07	Mon 19.3.07		Slavi, Jordan, Katia
9	Intermediate exam #1	1 day	Tue 20.3.07	Tue 20.3.07	8	Slavi, Jordan, Katia
10	Intro C++ (part 2)	10 days	Wed 21.3.07	Tue 03.4.07	9	Slavi, Katia
11	Final exam	1 day	Wed 04.4.07	Wed 04.4.07	10	Slavi, Katia
12	Prepare Core C++ materials	22 days	Tue 06.3.07	Wed 04.4.07		Katia[50%]
13	Interviews with the best candidates	4 days	Tue 10.4.07	Fri 13.4.07	7	EGT
14	**Core C++ Course**	**44 days**	**Mon 16.4.07**	**Thu 14.6.07**	**13**	**Slavi, Katia**
15	Prepare Core C++ materials	40 days	Mon 16.4.07	Fri 08.6.07		Katia[50%]
16	Core C++ (part 1)	7 days	Mon 16.4.07	Tue 24.4.07		Katia[50%];Slavi[50%]
17	Intermediate exam #1	1 day	Wed 25.4.07	Wed 25.4.07	16	Katia[50%];Slavi[50%]
18	Core C++ (part 2)	7 days	Thu 26.4.07	Fri 04.5.07	17	Katia[50%];Slavi[50%]
19	Intermediate exam #2	1 day	Mon 07.5.07	Mon 07.5.07	18	Katia[50%];Slavi[50%]
20	Core C++ (part 3)	7 days	Tue 08.5.07	Wed 16.5.07	19	Katia[50%];Slavi[50%]
21	Intermediate exam #3	1 day	Thu 17.5.07	Thu 17.5.07	20	Katia[50%];Slavi[50%]
22	Core C++ (part 4)	7 days	Fri 18.5.07	Mon 28.5.07	21	Katia[50%];Slavi[50%]
23	Intermediate exam #4	1 day	Tue 29.5.07	Tue 29.5.07	22	Katia[50%];Slavi[50%]
24	Practical project	12 days	Wed 30.5.07	Thu 14.6.07	23	Katia[50%];Slavi[50%]

Telerik Academy

Development Methodologies

Waterfall, Scrum, Lean Development, Kanban, Extreme Programming

What is a Development Methodology?

- A development methodology is a set of practices and procedures for organizing the software development process
  - A set of rules that developers have to follow
  - A set of conventions the organization decides to follow
  - A systematical, engineering approach for organizing and managing software projects

Development Methodologies

- Back in history
  - The "Waterfall" Process
    - Old-fashioned, not used today
  - Rational Unified Process (RUP)
  - Microsoft Solutions Framework (MSF)

- Modern development methodologies
  - Agile development processes
  - Scrum, Kanban, Lean Development, Extreme Programming (XP), etc.

Telerik Academy

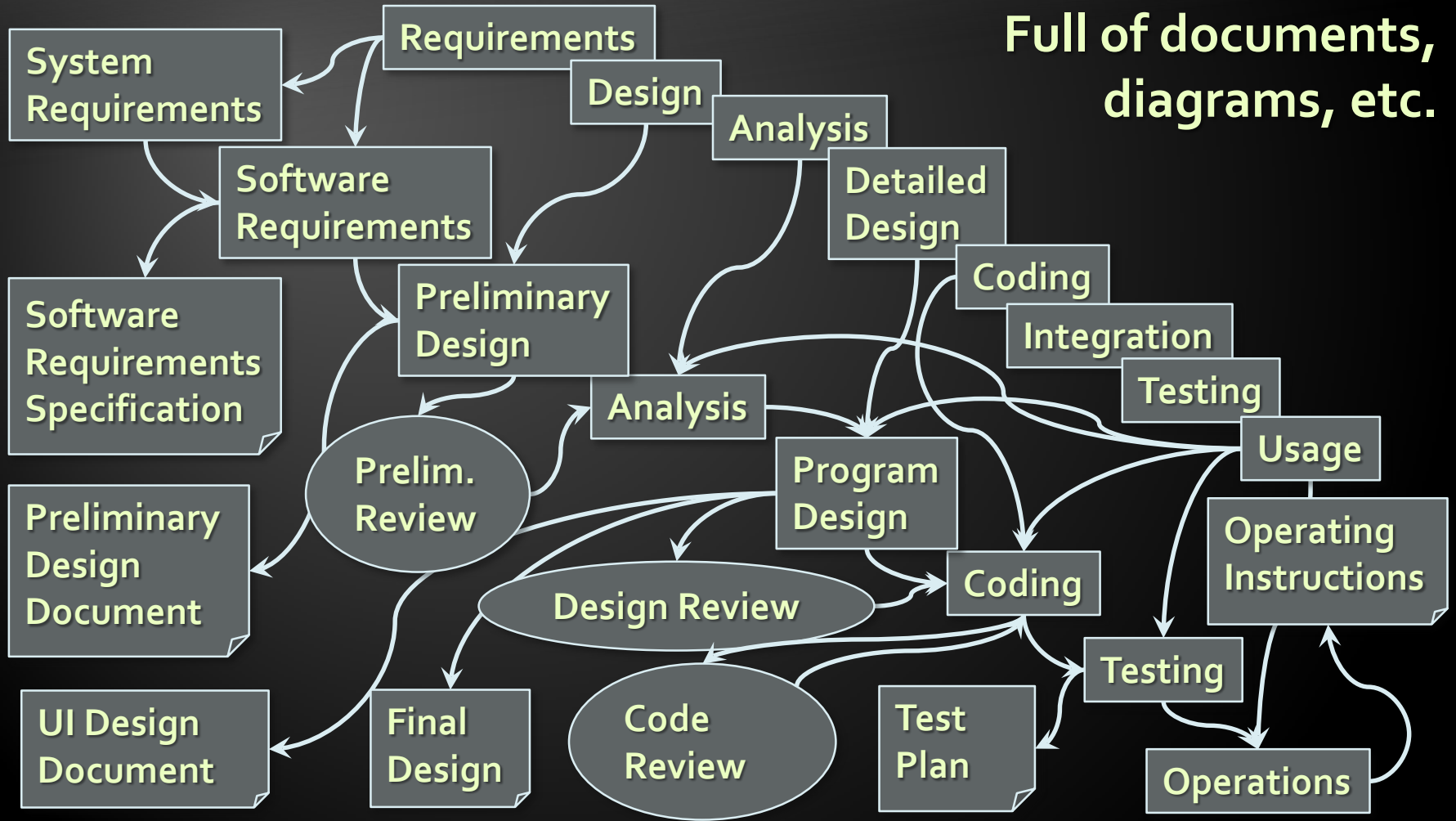
The Waterfall Development Process

The Waterfall Process

- The waterfall development process:

Formal Methodologies

- Formal methodologies are heavyweight!



Telerik Academy

Agile Development

The Agile Manifesto

"Our highest priority is to satisfy the customer through early and continuous delivery of valuable software"

Manifesto for Agile

The Agile Spirit

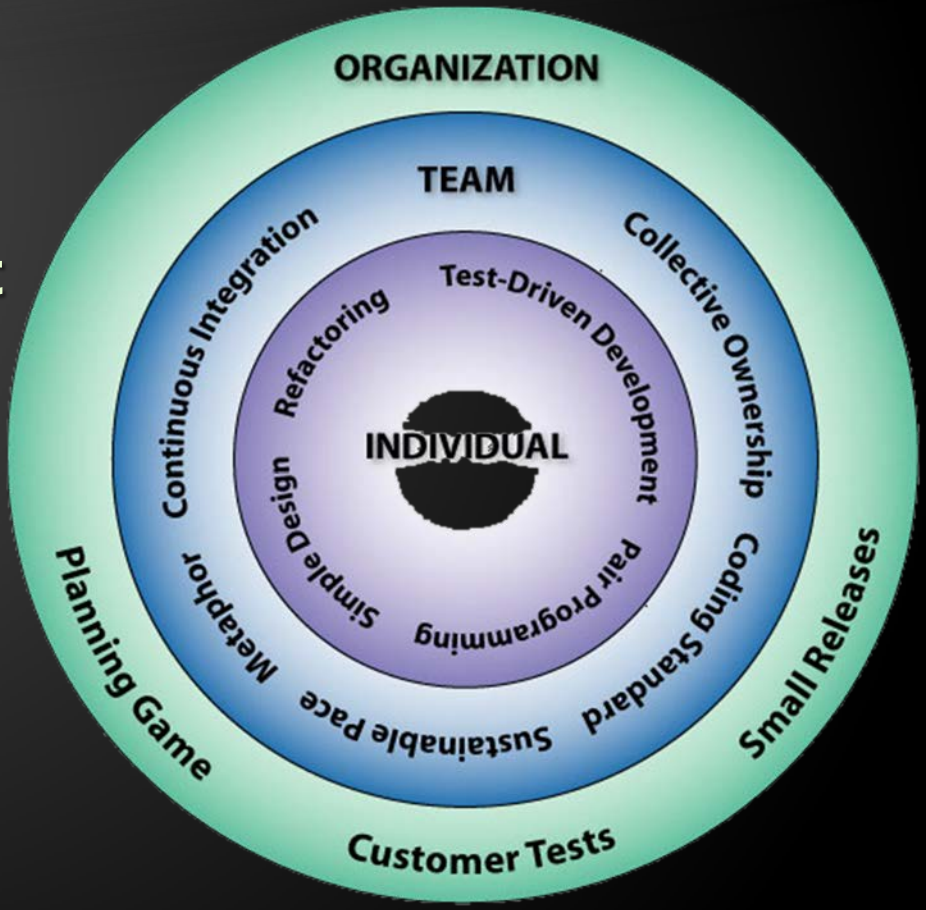
- Incremental
  - Working software over comprehensive documentation
- Cooperation
  - Customer collaboration over contract negotiation
- Straightforward
  - Individuals and interactions over processes and tools
- Adaptive
  - Responding to change over following a plan

Agile Methodologies

- Scrum
- Kanban
- Lean Software Development
- eXtreme Programming (XP)
- Feature-Driven Development (FDD)
- Crystal family of methodologies
- Adaptive Software Development (ASD)
- Dynamic System Development Model (DSDM)
- Agile Unified Process (AUP)

Extreme Programming: The 12 Key Practices

- The Planning Game
- Small Releases
- Metaphor
- Simple Design
- Test-Driven Development
- Refactoring
- Pair Programming
- Collective Ownership
- Continuous Integration
- 40-Hour Workweek
- On-site Customer
- Coding Standards



Scrum

- Scrum is an iterative incremental framework for managing complex projects

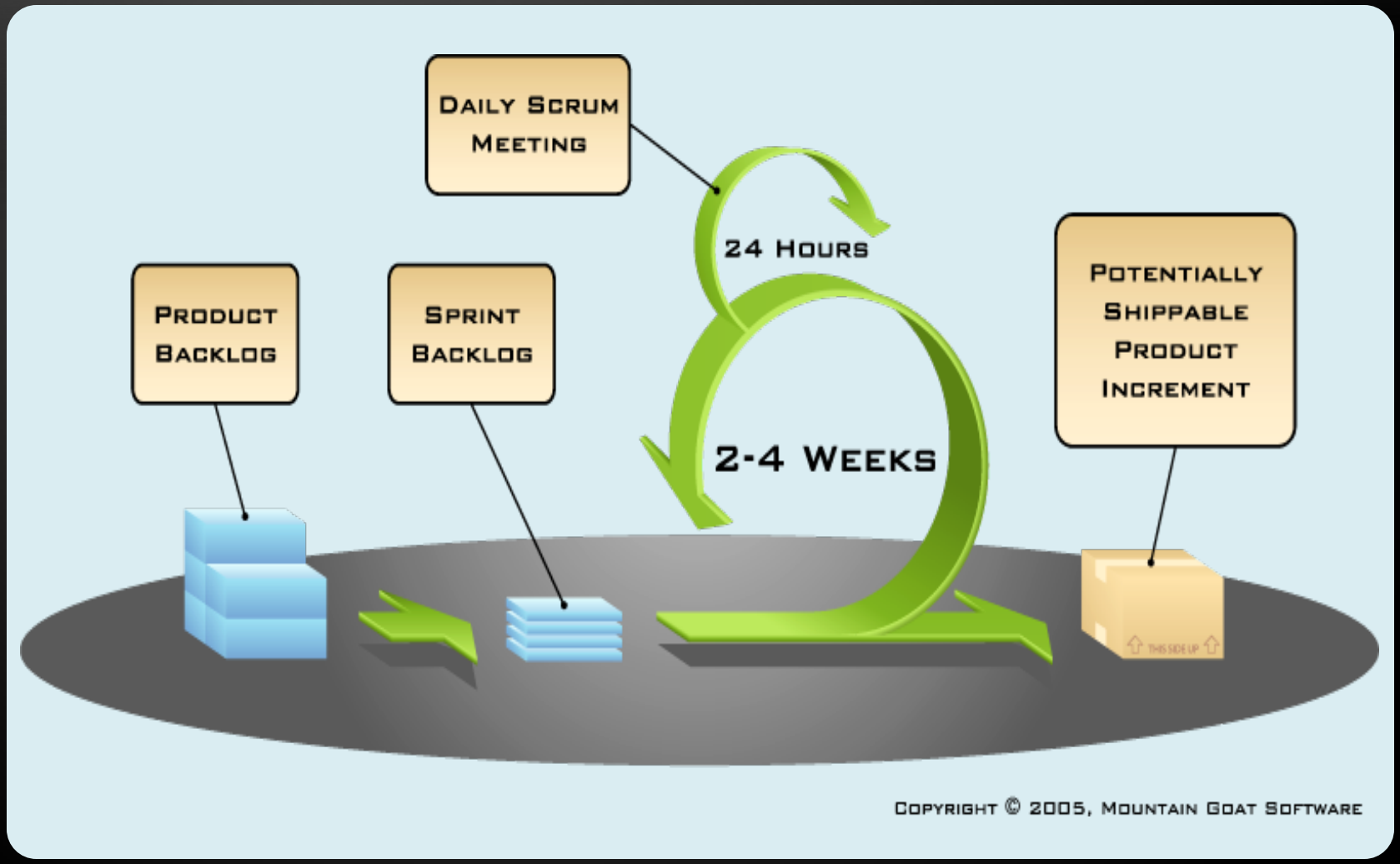
- Scrum roles:
  - Scrum Master – maintains the Scrum processes
  - Product Owner – represents the stakeholders
  - Team – a group of about 7 people
    - The team does the actual development: analysis, design, implementation, testing, etc.

Scrum Terminology

- Sprint
  - An iteration in the Scrum development
  - Usually few weeks

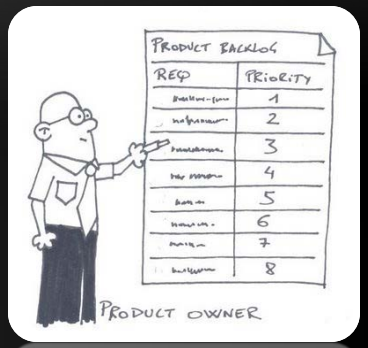
- Product Backlog
  - All features that have to be developed
- Sprint Backlog
  - All features planned for the current sprint

The Scrum Process Framework



Scrum Practices

- Sprint Planning Meeting
  - At the beginning of the sprint cycle
  - Establish the Sprint backlog



- Daily Scrum stand-up meeting
  - Each day during the sprint – project status from each team member
  - Timeboxed to 15 minutes
- Sprint Review Meeting
  - Review the work completed / not completed

Telerik Academy

Software Engineering Fundamentals

http://academy.telerik.com

Telerik Academy

Free Trainings @ Telerik Academy

- C# Programming @ Telerik Academy
  - csharpfundamentals.telerik.com

- Telerik Software Academy
  - academy.telerik.com

- Telerik Academy @ Facebook
  - facebook.com/TelerikAcademy

- Telerik Software Academy Forums
  - forums.academy.telerik.com