NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Software Engineering
Fundamentals

Svetlin Nakov

National Academy for Software Development

academy.devbg.org

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Agenda

1. Software engineering overview
    - Requirements
    - Design
    - Construction
    - Testing
    - Project management
2. Development methodologies overview
    - The Waterfall development process
    - Heavyweight methodologies
    - Agile methodologies and XP

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Software Engineering

Requirements, Design, Construction, Testing

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

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!
  - Often overlap and interact

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Software Requirements

Functional, Non-functional Requirements, SRS

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**
- ***Prototyping*** **is often used, especially for the user interface**
- **The outcome is the Software Requirements Specification (SRS)**

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, etc.

Software Requirements

- It is always hard to describe and document the requirements in comprehensive and not ambiguous way
  - Good requirements save time and money
- Requirements always change during the project!
  - Good software requirements specification reduces the changes
  - Prototypes significantly reduce changes

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Software Requirements Specification and UI Prototype – Examples

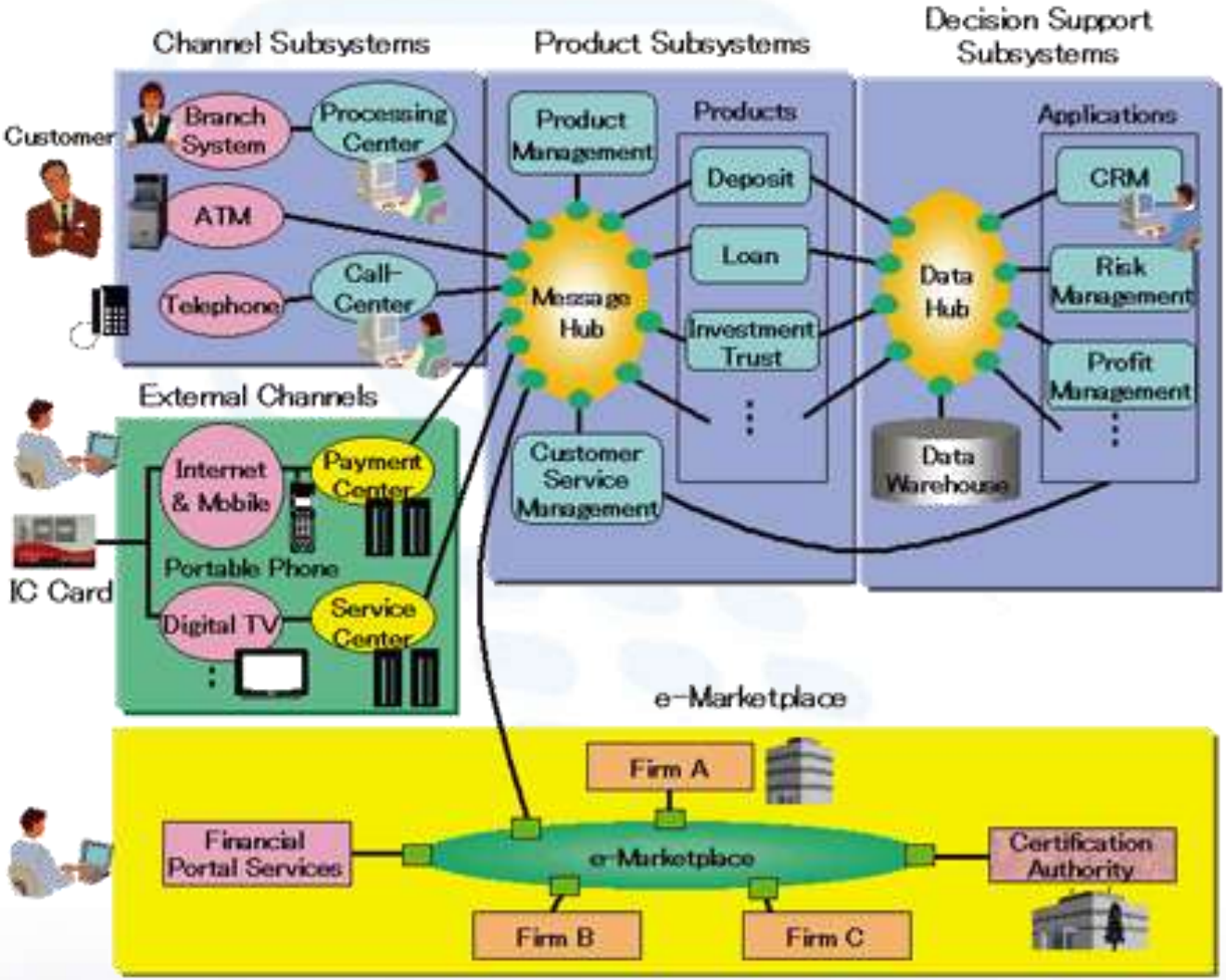
NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Software Architecture and Software Design

Software Architecture and Software Design

- ***Software design*** **is a technical description 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 modules**
  - **Platforms and technologies**

System 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)*** is a formal description of the architecture and design of the system
- It contains:
  - Architecture design
    - Modules and their interaction (diagram)
  - For each module
    - Process design (diagrams)
    - Data design (E/R diagram)
    - Interfaces design (class diagram)

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Software Design Document – Example

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

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 code
  - Writing unit tests (sometimes)
  - 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 write it**
  - **At least run it to see the results**
  - ***Unit testing*** **is even better**
    - **Units tests can be repeated many times**
- **System testing is done by QA engineers**
  - **Unit testing is done by developers**

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

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 code causing it**
  - **Fix the defect**
  - **Test to check if the fix is correct**

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Integration

- ***Integration*** **is putting all pieces together**
  - **Compile, run and deploy the modules as 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***

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Software Verification and Testing

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Software Verification

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

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

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

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

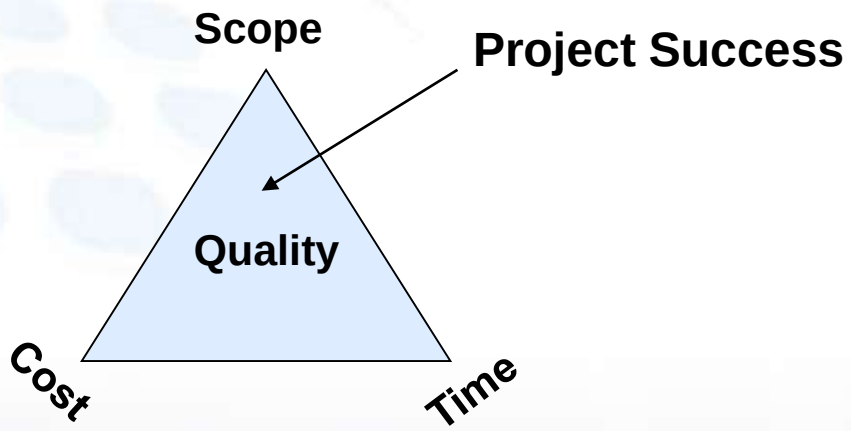
Test Plans and Test Cases – Example

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Software Project Management

What is Project Management?

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



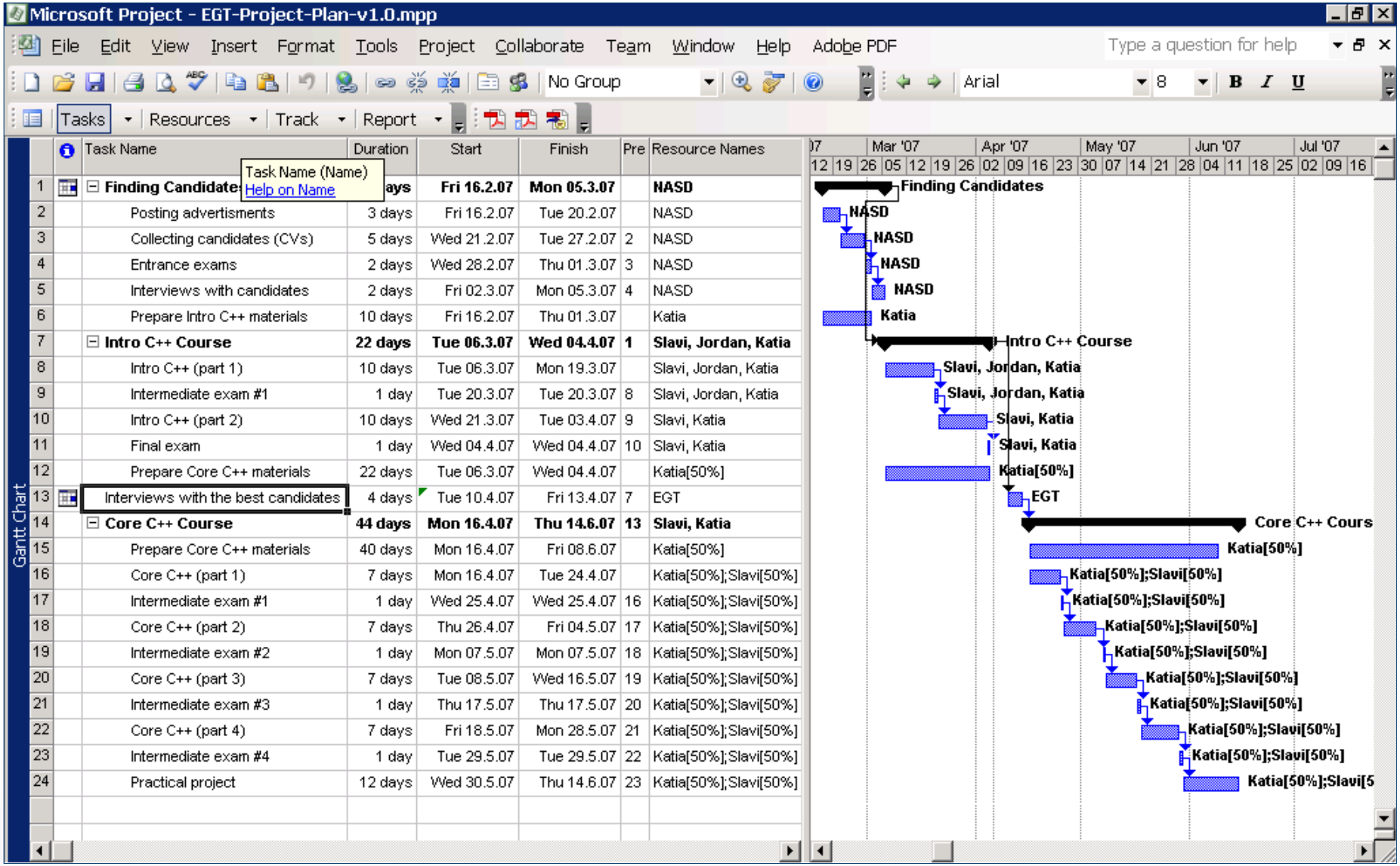
What is Software Project Management?

- ***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, etc.
- Project management tools simplify creating and monitoring project plans

Project Plan – Example



NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Development Methodologies

What is a Development Methodology?

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

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Development Methodologies

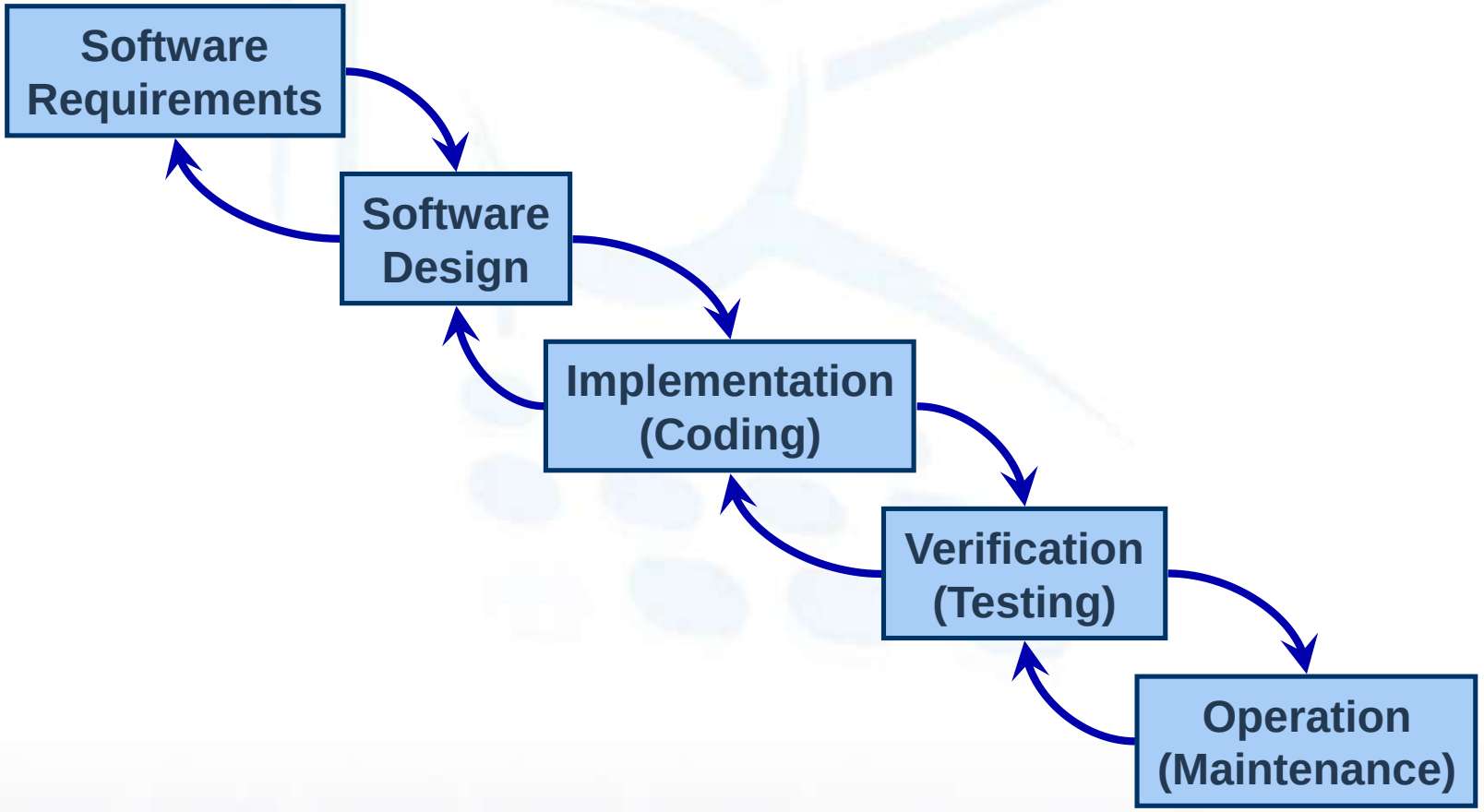
- The "Waterfall" Process
  - Old-fashioned, not used today
- Rational Unified Process (RUP)
  - Very formal, lots of documentation
- Microsoft Solutions Framework (MSF)
  - Formal heavyweight approach
- Agile Development Processes
  - E.g. Extreme Programming

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

The Waterfall Development Process

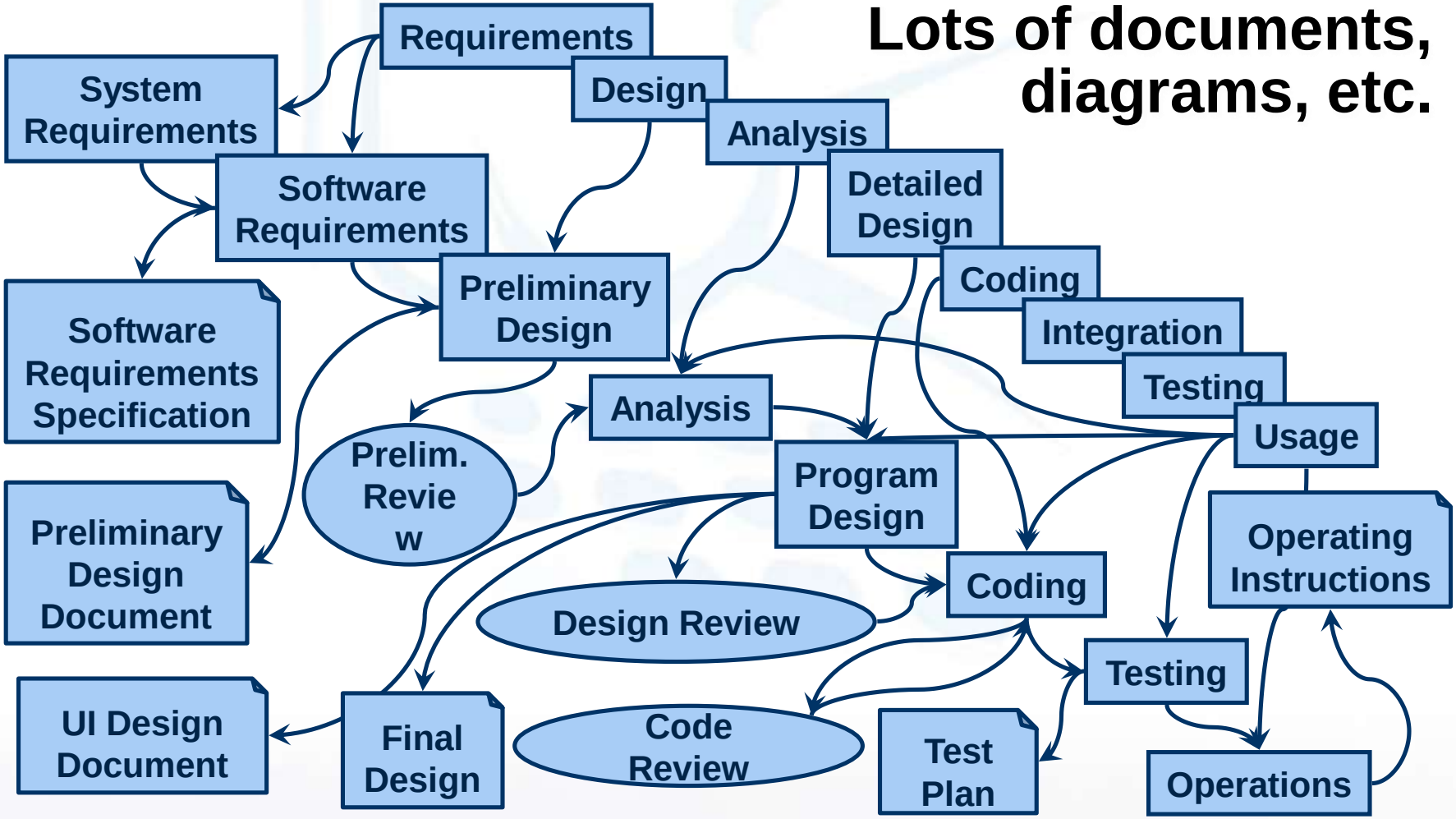
The Waterfall Process

- **The waterfall development process:**



Formal Methodologies

- **Formal methodologies are heavyweight!**



NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Agile Development

The Agile Manifesto

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

Manifesto for Agile

NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

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

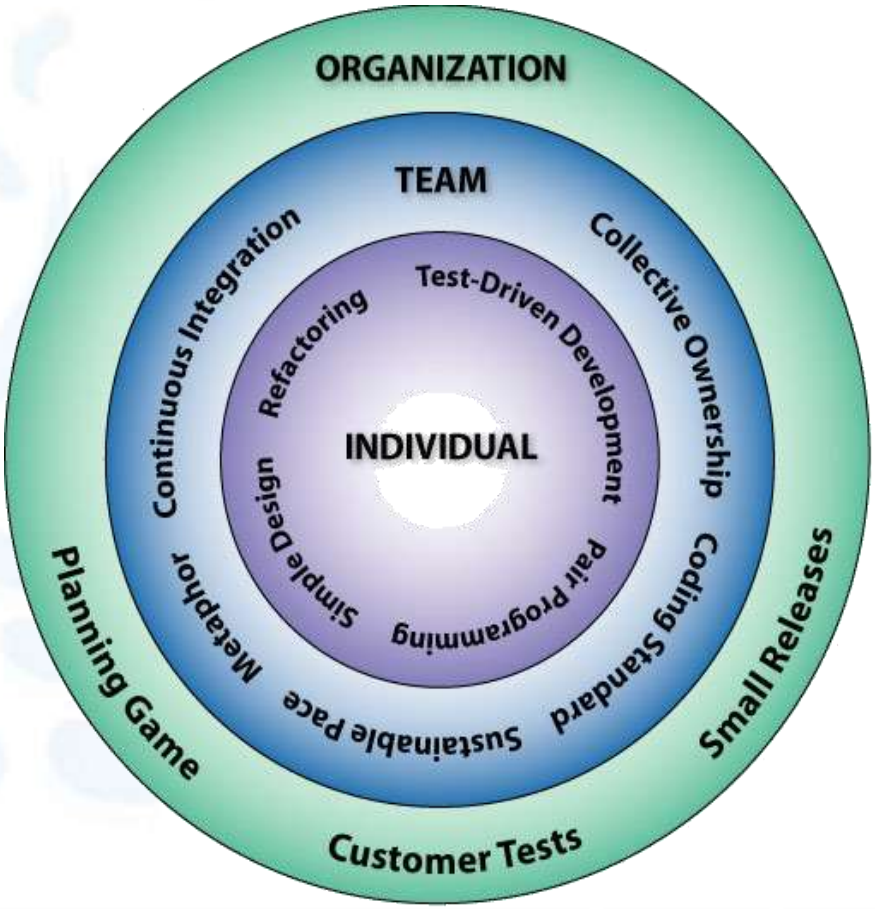
NASD
NATIONAL ACADEMY FOR SOFTWARE DEVELOPMENT

Agile Methodologies

- eXtreme Programming (XP)
- Scrum
- Crystal family of methodologies
- Feature-Driven Development (FDD)
- 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



Questions?