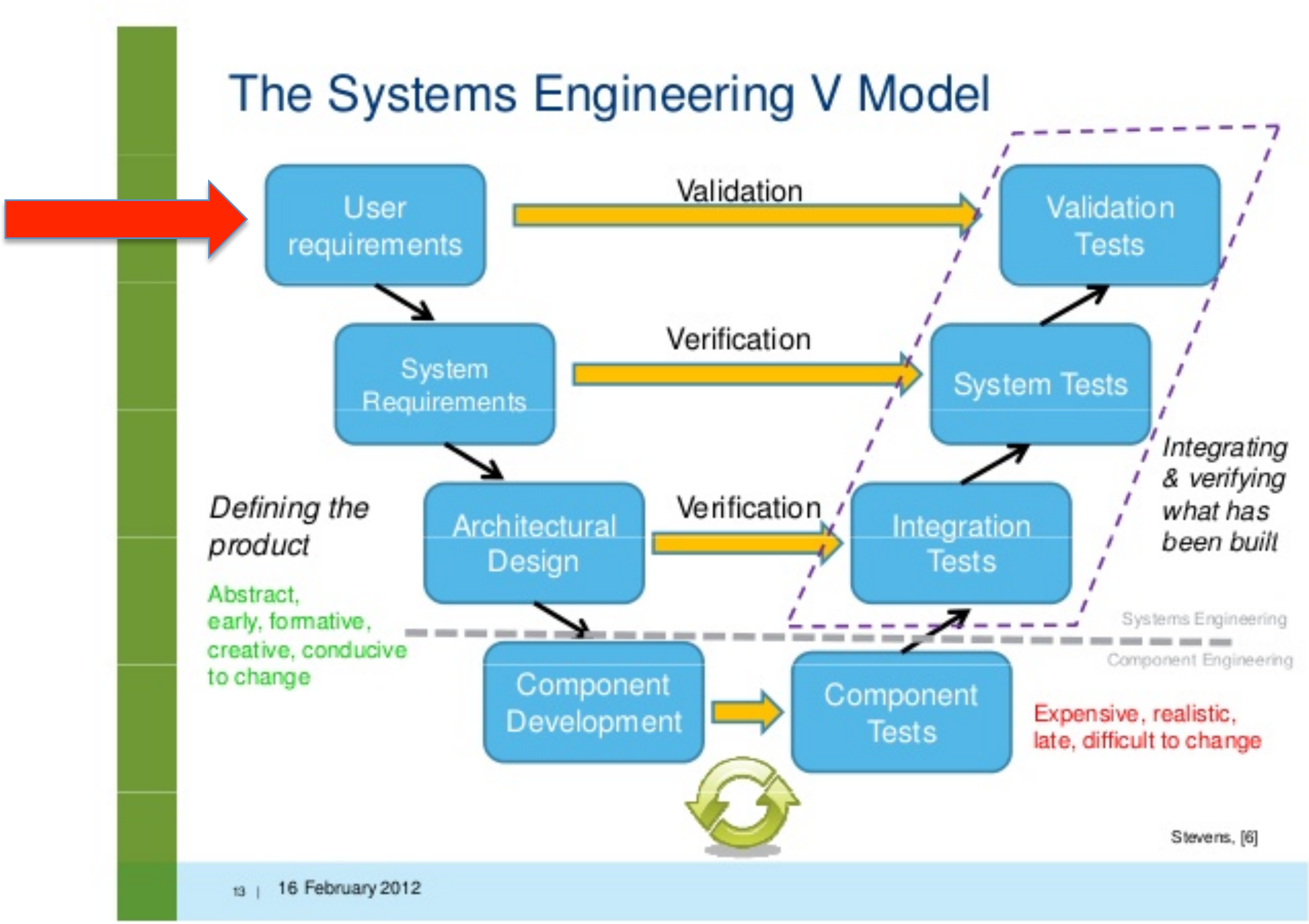
Requirements development

Jim Kowalkowski

Reminder: Requirements management

- “The purpose of requirements management is to ensure that an organization documents, verifies, and meets the needs and expectations of its customers and internal or external stakeholders” – Wikipedia
- “Requirements Management helps ensure that the end product meets the needs and expectations of the stakeholders.” - CDC Unified Process Practices Guide
- IEEE definition of a requirement:
  - (1) a condition or capability needed by a user to solve a problem or achieve an objective
  - (2) a condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard,specification, or other formally imposed document
  - (3) a documented representation of a condition or capability as in (1) or (2)
- Requirements provide:
  - Common understanding
  - Information to validate system
  - Inform project planning and scheduling
  - Provide traceability

The V diagram



The Systems Engineering V Model

User requirements

Validation

Validation Tests

System Requirements

Verification

System Tests

Architectural Design

Verification

Integration Tests

Component Development

Component Tests

Defining the product

Abstract, early, formative, creative, conducive to change

Integrating & verifying what has been built

Systems Engineering

Component Engineering

Expensive, realistic, late, difficult to change

Stevens, [6]

13 | 16 February 2012

Requirement categories

- **Functional requirements** are the fundamental or essential subject matter of the product. They describe what the product has to do or what processing actions it is to take.
- **Nonfunctional requirements** are the properties that the functions must have, such as performance and usability. Do not be deterred by the unfortunate type name (we use it because it is the most common way of referring to these types of requirements)—these requirements are as important as the functional requirements for the product's success.
- **Design constraints** impose restrictions on how the product must be designed. For example, it might have to be implemented in the hand-held device being given to major customers, or it might have to use the existing servers and desktop computers, or any other hardware, software, or business practice.

Source: Volere template

Requirement types

- **User Requirements** define the results the users expect from the system
  - “The homeowner shall hear an alarm when smoke is detected.”
- **System Requirements** define what the system must do to satisfy the users
  - “The alarm will produce a sound between 125 – 155 dBA.”
- **Design Requirements** define all of the components necessary to achieve the system requirements
  - “The alarm will be produced by part # 123-45-678.”

Source: Telelogic

Comparison of User and system requirements

User	System
Description of the problem	Abstract solution
In user language	In developer language
Organized by goals	Organized by functions in a hierarchy
Subject is a type of user	Subject is the system (or subsystem)
Defines what the user gets	Defines what the system does
Owned by users	Owned by developers

Source: Telelogic

Use cases

- What is a use case?
  - “A use case captures a contract between the stakeholders of a system about its behavior. The use case describes the system’s behavior under various conditions as it responds to a request from one of the stakeholders, called the primary actor. The primary actor initiates an interaction with the system to accomplish some goal. The system responds, protecting the interests of all the stakeholders.” - Alistair Cockburn
- In others word, a use cases captures the interactions between a user and the system for a particular scenario.
  - Example: “submit a job for processing using the available batch resources”
- Use cases have been used successfully to solicit or expose functional requirements.

Importance of a glossary

- The glossary contains agreed-upon definitions of the words and phases used in the requirements
- This will help make requirement statements concise and clean.

Use of roles

- It is useful during the collection process to have participants take on a particular role.
  - Helps focus the discussion
- Identify the role that a requirements applies to

A useful set of roles

- Primary Roles
  - Algorithm Developer. A software developer (scientist, computer scientist, computing professional) with specific expertise in developing and implementing algorithms for processing experimental data.
  - Algorithm tester/validator. Someone who makes sure the algorithms are working correctly and fit well into the system (performance, conformance, maintenance for example).
  - Scientist Data Analyst. A scientist who uses the systems to perform scientific analyses of data.
- Secondary Roles
  - External Package Provider. Provider of an external package that interfaces to and depends on or is depended upon by this software system.
  - Infrastructure Developer. A developer of the computing, system and framework infrastructure.
  - Overseer. Individual who is sponsoring, reviewing, reporting on the system's functionality and status.
  - Production Processing Operator. Individual who runs the reconstruction systems for production data processing.
  - Scientist Configuration Developer. Person that designs and consults on the configurations that would be utilized by the operators and users.

Requirements structure

- Form a hierarchy
  - The V diagram shows the essential *levels* on the left side
- Relate to testing
  - The V diagram also shows this on the right side
  - Each requirement ought to have a way to verify or validate whether or not is satisfied
- Each level has one or more documents
  - Grouping the requirements by related topic or role is constructive.

Requirements writing advice from Telelogic

- Complete sentence
- States subject and predicate
  - Subject is a user type or the system under consideration
  - Predicate is a condition, action, or intended result
- Consistent use of language
- Specifies:
  - Desired goal or result (User requirement)
  - Function (System requirement)
  - Constraint (either)
- Contains a success criterion or other measurable indication of the quality

The challenge is to seek out the user type, end result, and success measure in every requirement you define.

http://www.slideshare.net/LizLavaveshkul/writing-effective-requirements-41453341

Examples

- Algorithm developer must achieve e/γ separation of 90% efficiency for electrons
- Analysts must have separate, *run-time configurable* steps in the *reconstruction* chain for *Pattern recognition* and *track fitting*.
- Reconstruction algorithm developers must have relevant pedestal calibration datasets accessible
- Data analysts must have *Calibration datasets* accessible to algorithms from *off-network resources*
- Algorithm developers must have access to *side-band* (data from previous and future events) from within the currently active event.

Requirement form

- Fully formed requirements have several useful attributes filled in. A useful set of attributes is
  - ID: short string label, a sequence number plus area it is under.
  - Description: short text using terms used should be from glossary.
  - Parent ID: requirements that caused this requirement to exist.
  - Requirement type: Science, Software, System, Human, Interface, Business.
  - Priority - high, medium, low, wish
  - Value goal - number or range
  - Units - the units of the value goal
  - Author(s) - Person(s) inserting or modifying this requirement
  - Verification method - Data Analysis, Demonstration, Inspection, Simulation, Test/Measure
  - Time needed - <6 months; 6 months to 24 months; >24 months
  - Status - Proposed, accepted, validated, obsolete, superseded
  - External reference(s) - background or other material relevant for this requirement.
  - Comments - Any additional information that will help others to best understand this requirement, including examples from what currently exists.