Specification & Documentation

Starting from elicited & evaluated requirements

- Precise definition of all features of the agreed system
  - Objectives, concepts, relevant domain properties, system/software requirements, assumptions, responsibilities
  - Rationale for options taken, satisfaction arguments
  - Likely system evolutions & variants
- Organization of these in a coherent structure
- Documentation in a form understandable by all parties
  - Often in annex: costs, workplan, delivery schedules

Resulting product: Requirements Document (RD)

Documentation Methods

Natural Language

- Free documentation in unrestricted natural language
- Disciplined documentation in structured natural language
  - Local rules on writing statements
  - Global rules on organizing the Requirements Document

Use of diagrammatic notations

- System scope: Context, problem, frame diagrams
- Conceptual structures: Entity-relationship diagrams
- Activities and data: SADT diagrams
- Information flows: Dataflow diagrams
- System operations: Use case diagrams
- Interaction scenarios: Event trace diagrams
- System behaviours: State machine diagrams
- Stimuli and responses: R-net diagrams
- Integrating multiple system views, multi-view spec in UML

Formal specification

- Logic as a basis for formalizing statements
- State-based specification
- Algebraic specification
- History-based specification
- Event-based specification
- Ontology-based specification (Domain Model)

Use of diagrammatic notations

Alleviate problems stemming from NL specifications

- Diagrams used to complement or replace NL prose
- Dedicated to specific aspects of the system (as-is or to-be)
- Graphical: to ease communication, provide overview
- Semi-formal: tool support, machine-processable

Pros: Graphical Representation

- overview & structuring of important aspects
- easy to understand, communicate
- surface-level analysis, supported by tools (e.g., query engines)

Cons: Only Semi-Formal Specification

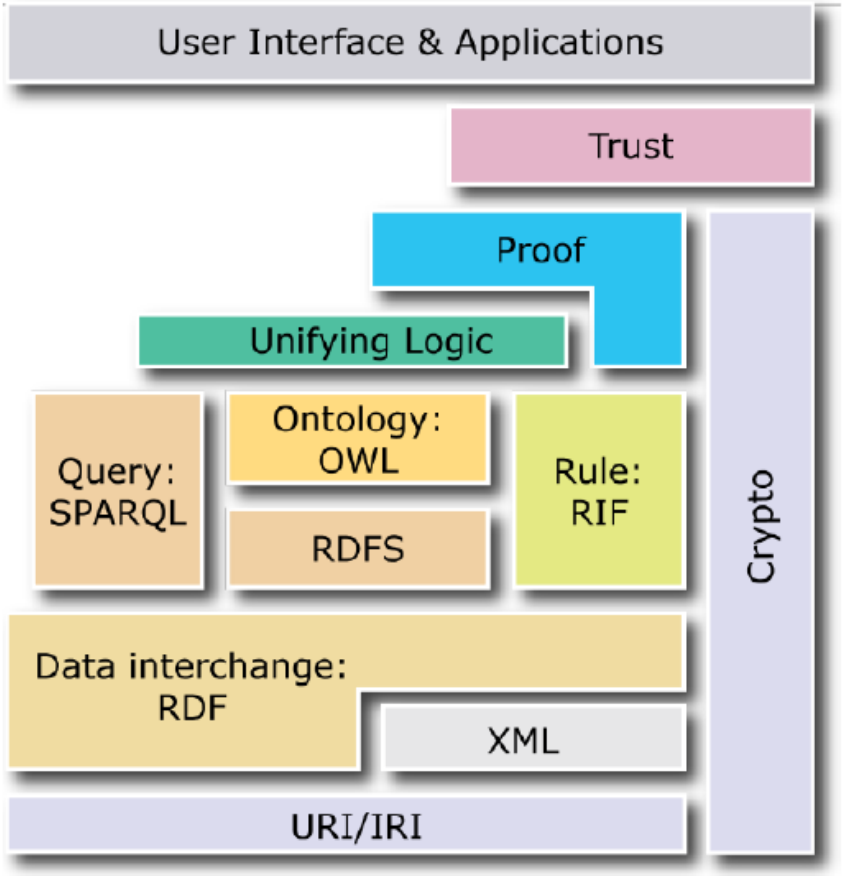
- language semantics may be vague (different interpretations)
- only surface-level aspects formalized, not item properties
- limited forms of analysis

Multi-view specification in UML

UML Diagrams

The Unified Modeling Language (UML) has standardized notations for diagrams relevant to RE

Modern Approach: Use of W3C standards (RDF, RDFS, OWL) in RE/SE



Unrestricted Natural Language

Defects & Errors in Natural Language Specifications

Free Documentation in Unrestricted Natural Language

Unconstrained prose writing in natural language (NL): : :

_ Unlimited expressiveness, communicability, no training needed

_ Prone to many of the specification errors & flaws

Challenge in NL: Ambiguities

_ Ambiguities are inherent to NL; these can be harmful:

"Full braking shall be activated by any train that receives an outdated acceleration command or that enters a station block at speed higher than X m.p.h. and for which the preceding train is closer than Y yards."

_ Frequent confusions among logical connectives in NL

For example, case analysis:

if Case1 then <Statement1>
or if Case2 then <Statement2>

(amounts to true!) vs.

if Case1 then <Statement1>
and if Case2 then <Statement2>

"Mary had a little lamb"

Analysis in [LW03, Chapter 23]

In [LW03], an analysis shows how to derive > 10 different meanings (semantic interpretations) of "Mary had a little lamb", from

Mary owned a little sheep under one year of age or without permanent teeth.

to

Mary bribed a small person trading in securities who was easily cheated.

Practical Consequences

Reduce ambiguity through writing guidelines (beyond the glossary)

_ Challenge: finding "sweet spot" between readability and ambiguity

_ Increased importance for international projects (outsourcing, offshoring, distributed teams etc.), due to differrent cultural background of team members

Defects in Requirements Documents

Wide range of possible errors & flaws

- _ Omission (critical error!)
- _ Contradiction (critical error!)
- _ Inadequacy (critical error!)
- _ Ambiguity (critical error!)
- _ Unmeasurability
- _ Noise
- _ Overs pecification
- _ Unfea sibility (wishful thinking)
- _ Unintelligibility
- _ Poor structuring
- _ Forward reference
- _ Remorse
- _ Opacity

Errors in a requirements document

Omission: problem world feature not stated by any RD item
- _ e.g., no req about state of train doors in case of emergency stop

Contradiction: RD items stating a problem world feature in an incompatible way
- _ “Doors must always be kept closed between platforms” and
- _ “Doors must be opened in case of emergency stop”

Inadequacy: RD item not adequately stating a problem world feature
- _ “Panels inside trains shall display all flights served at next stop”

Ambiguity: RD item allowing a problem world feature to be interpreted in different ways
- “Doors shall be open as soon as the train is stopped at platform”

Unmeasurability: RD item stating a problem world feature in a way precluding option comparison or solution testing
- _ “Panels inside trains shall be user-friendly”

Defects in a requirements document

Noise: RD item yielding no information on any problem world feature
(Variant: uncontrolled redundancy)
- _ “Non-smoking signs shall be posted on train windows”

Overspecification: RD item stating a feature not in the problem world, but in the machine solution
- _ “The setAlarm method shall be invoked on receipt of an Alarm message”

Unfeasibility: RD item not implementable within budget/schedule
- _ “In-train panels shall display all delayed flights at next stop”

Unintelligibility: RD item incomprehensible to those needing to use it
- _ A requirement statement containing 5 acronyms

Poor structuring: RD item not organized according to any sensible & visible structuring rule
- _ Intertwining of acceleration control and train tracking issues

Forward reference: RD item making use of problem world features not defined yet
- _ Multiple uses of the concept of worst-case stopping distance before its definition appears several pages after in the RD

Remorse: RD item stating a problem world feature lately or incidentally
- _ After multiple uses of the undefined concept of worst-case stopping distance, the last one directly followed by an incidental definition between parentheses

Poor modifiability: RD items whose changes must be propagated throughout the RD
- _ Use of fixed numerical values for quantities subject to change

Opacity: RD item whose rationale, authoring or dependencies are invisible
- _ “The commanded train speed must always be at least 7 mph above physical speed” without any explanation of rationale for this

<3> **Structured Natural Language**

Disciplined documentation in structured Natural Language
Decision Tables
Using Templates

Disciplined documentation in structured NL

Local rules on writing statements

Use stylistic rules for good NL spec, e.g.:

_ Identify who will read this; write accordingly
_ Say what you are going to do before doing it
_ Motivate first, summarize after
_ Make sure every concept is defined before use
_ Keep asking yourself: “Is this comprehensible? Is this enough? Is this relevant?”
_ Never more than one req, assumption, or dom prop in a single sentence. Keep sentences short.
_ Use “shall” for mandatory, “should” for desirable prescriptions
_ Avoid unnecessary jargon & acronyms
_ Use suggestive examples to clarify abstract statements
_ Supply diagrams for complex relationships among items (more in the van Lamsweerde)

Other Criteria

Passive Voice

In general, avoid descriptions in passive voice.

_ With passive voice, the agent (who is performing the action?) is missing
BAD: “The order is entered.”
BETTER: “The user enters the order.”

User’s Perspective

Write User Requirements from User’s Point of View!
Example: Login Validation
BAD System shall prompt for user identification and password.
BETTER User shall start accessing the system by identifying himself.

Beyond NL Specifications

Natural Language alone can be too limiting

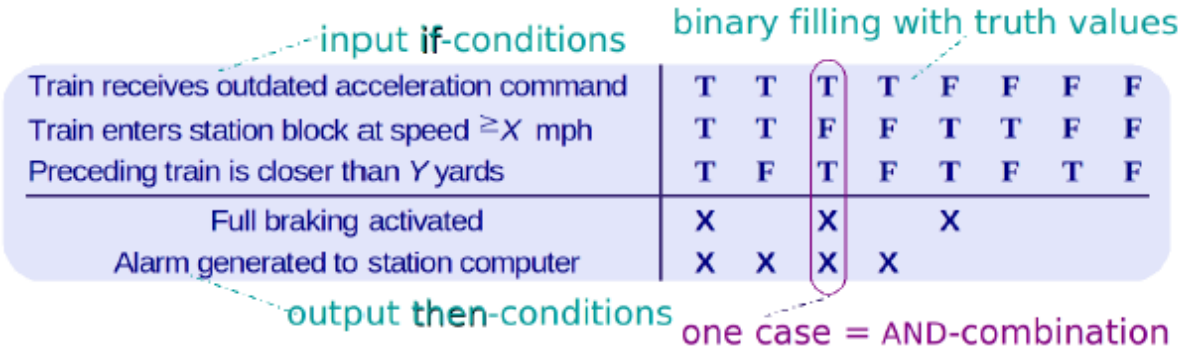
_ Some facts (e.g., complex business rules) cannot be expressed easily in an unambiguous fashion using NL
_ Solution: Complement NL spec with other techniques, e.g., Decision Tables

Example from Aviation

“The Landing Pilot is the Non-Handling Pilot until the ’decision altitude’ call, when the Handling Non-Landing Pilot hands the handling to the Non-Handling Landing Pilot, unless the latter calls ’go-around,’ in which case the Handling Non-Landing Pilot continues handling and the Non-Handling Landing Pilot continues non-handling until the next call of ’land’ or ’go-around’ as appropriate. In view of recent confusions over these rules, it was deemed necessary to restate them clearly.” (British Airways memorandum, quoted in Pilot Magazine, December 1996)

Decision Tables

Use decision tables for complex combinations of conditions:



Systematic, simple, additional benefits

_ Completeness check: 2^N columns required for full table
_ Table reduction:
_ drop impossible cases in view of domain properties;
_ merge 2 columns differing only by single “T”, “F”) “-”
_ Test cases for free (cause-effect coverage)

Local rules on writing statements (3)======USING TEMPLATES

Use standardized statement templates

Identifier suggestive; hierarchical if compound statement
Category functional or quality req, assumption, domain property, definition, scenario example, : :
Specification statement formulation according to stylistic rules
Fit criterion for measurability (see next slide)
Source for traceability to elicitation sources
Rationale for better understanding & traceability
Interaction contribution to, conflict with other statements
Priority level for comparison & prioritization
Stability, Commonality levels for change management

Fit Criteria

Fit criteria make statements measurable

_ Complement statements by quantifying the extent to which they must be satisfied [RR12]
_ Especially important for measurability of NFRs

Examples

Spec: The scheduled meeting dates shall be convenient to participants
Fit criterion: Scheduled dates should fit the diary constraints of at least 90% of invited participants in at least 80% of cases

Spec: Info displays inside trains shall be informative & understandable
Fit criterion: A survey after 3 months of use should reveal that at least 75% of travelers found in-train info displays helpful for finding their connection

<4> Global Rules and Standards

Requirements Classification
IEEE Std. 830-1998
Unified Process (UP)

Global rules on organizing the RD

Grouping rules:

Put in same section all items related to common factor
_ system objective
_ system component
_ task
_ conceptual object
_ software feature _ : : :

Templates

Global templates for standardizing the RD structure
_ domain-specific, organization-specific, company-specific

Requirements Classification

Taxonomies

Unfortunately, there is no single, standardized categorization for requirements
_ Definitions different by author, standard, etc.

Typical Variations

However, most differentiate in functional and "something else", which is typically a single or combination of
_ Non-Functional Requirement
_ Quality Requirement
_ (Design) Constraint
In practise, it is important to check the definition used by your company/standard/textbook/etc. to avoid improper classification.

Functional Requirements

Prescribe what services the software-to-be should provide
_ capture intended software effects on environment, applicability conditions
_ units of functionality resulting from software operations

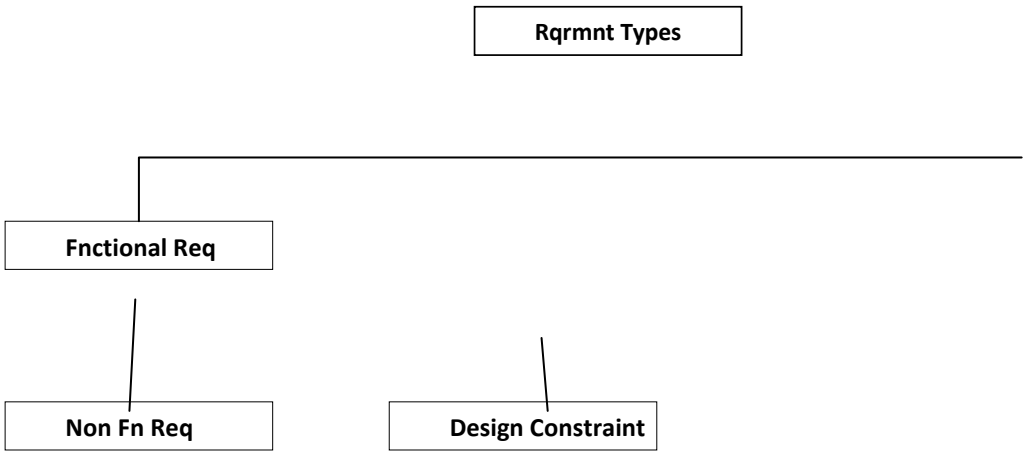
Non-Functional Requirements

Constrain how such services should be provided
_ Quality requirements: safety, security, accuracy, time/space performance, usability, : : :
_ Other requirements: compliance, architectural, development reqs

FR, NFR, DC (Leffingwell/Widrig)

Design Constraint

Roughly corresponds to the constraints in the NFR taxonomy by van Lamsweerde [vL09].



FURPS+

FURPS+ Classification [Grady, HP]

FURPS+ is often used in UP projects:

- Functional: Functional requirements
- Usability: Human factors, help, documentation, : : :
- Reliability: Frequency of failure, recoverability, predictability, : : :
- Performance: Response time, throughput, accuracy, : : :
- Supportability: Adaptability, maintainability, configurability, : : :

With the + standing for additional constraints:

- _ Design Constraints
- _ Interface Constraints
- _ Implementation Constraints
- _ Operation Constraints
- _ Legal Constraints:::

NFRs

The system shall support up to 2000 simultaneous users against the central database at any given time, and up to 500 simultaneous users against the local servers at any one time. The system shall provide access to the legacy course catalogue database with no more than a 10 second latency.

Design Constraint

The system shall integrate with existing legacy system (course catalogue database) which operates on the College DEC VAX Main Frame.

IEEE Std 830 template for organizing the RD SEE SLIDE 33 / 34

Variant: (commercial) VOLERE template [RR12]:
_ explicit sections for domain properties, costs, risks, development workplan, : : :

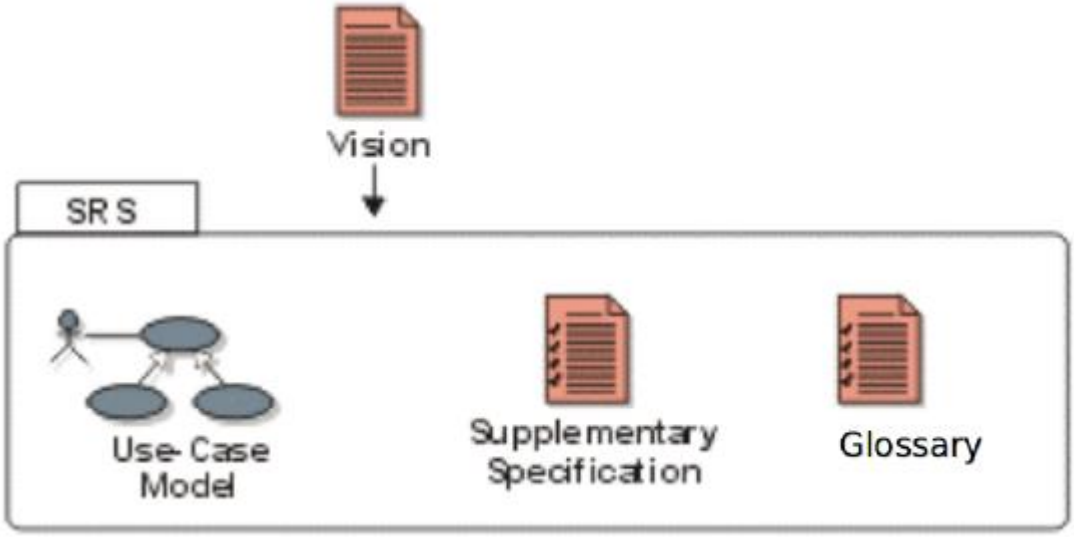
IEEE Std 830: Quality

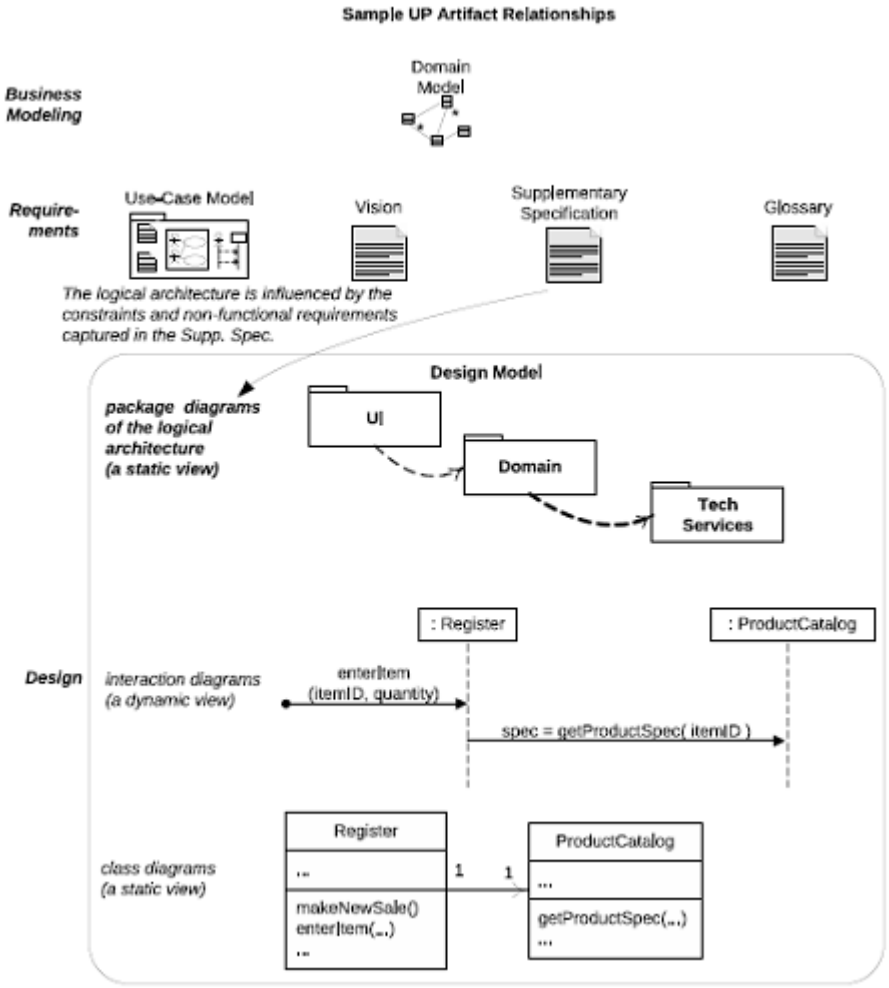
Characteristics of good SRS:			**An SrS should Be:**
Correct	**Complete**	**Consistent**	**Verifiable**
Modifiable	**Unambiguous**	**Traceable**	**Ranked for importance &/or stability**

Unified Process (UP) Requirements Artifacts

Artifact Types used in UP

- Vision Document: High-level requirements
- Use Case Model: Functional requirements, expressed in scenarios
- Supplementary Specification: Functional requirements, Non-functional requirements, design constraints
- Glossary: Terms and definitions





Supplementary Specification: Content

Functional requirements not expressed in Use Cases

E.g., Reports, Logging, Computations, : : :

Non-functional requirements

_ Quality attributes of the system
_ Legal, regulatory and documentation requirements

Design constraints

Interfaces

User Interfaces, Hardware Interfaces, Software Interfaces, Communication Interfaces

Domain Business Rules

For narrow application-specific rules; global domain rules (e.g., laws, regulations)

go into the UP Business Rules artifact.

Reading Material

Required

_ [vL09, Chapter 4–4.2] (Requirements Specification and Documentation)
_ [LW03, Chapter 22] (Supplementary Specification)
_ [LW03, Chapter 23] (Ambiguity and Specificity)
_ [IEE98, Section 5 & Annex A] (IEEE Standard 830-1998)