RD Ontology

This is an ontology for requirements and their development.

To understand the concept of "ontology" [see reference 4.1 at http://understandingrequirements.com/Chapter-4-Reference.php], consider the application domain of "hotels". This domain is associated with objects (rooms, furnishings, guests) having properties with values (types and number of beds) and relationships between objects (beds in a room, adjacent rooms). Domains are also associated with facts (each room has a microwave), actions (renting an available room), and behavior rules (When an available room is rented, the room becomes "rented"). This collection of associated concepts is called a domain "ontology".

Consider three airline support systems: a reservation system, an aircraft and crew scheduling system, and an aircraft maintenance system. Each system needs different aspects of the same domain, i.e., different objects (such as customers, crew, and parts) and different attributes of common objects (such as aircraft). Each system needs a "subdomain ontology". The reservation system needs available seats, the scheduling system needs crew requirements and current aircraft location, and the maintenance system needs use and maintenance history.

An ontology may be partially specified in a glossary.

Vision statement

A vision statement captures the end user's or customer's ideas and views of the system or application to be developed. It may have an associated problem or opportunity description.

Goals

Primary features

Includes capabilities, functions, and attributes

Application context

Stakeholders

Customers

Users, including operators

Developers

Domain experts

Safety engineers

Quality & Productivity staff

Project managers

Product managers

Configuration managers

Application domain ontology

Domain objects with properties

Object relationships

Domain facts

Domain actions and behavior rules

Other domain applications

Includes their interfaces.

System utilities

Assumptions

Includes operating environments

Requirements - details of a vision

A (real) **requirement** is a restriction that determines whether a proposed solution is adequate. If a proposed solution complies with a requirement, the solution is adequate relative to that requirement. A proposed solution may comply with some requirements, but not others. A proposed solution may comply with all known requirements, but still be inadequate because (1) it does not comply with some missing, but critical requirements or because (2) it is unachievable.

Individual requirements have the following properties:

- identifier
- type
- primary sources
- inherent understanding risk level
- priority level
- states: < Incomplete, Complete, Validated, Implemented, Retired>

Hazard mitigations

The results of hazard analysis should be turned into requirements by associating each serious hazard with its set of mitigations.

Quality goals

A 3D Whizfolders model of quality attributes is freely downloadable from
http://www.quality-aware.com/software-quality-KB.php

Internal qualities

primary sources = quality management
inherent understanding risk level = low

For a detailed description, see the link to the 3D Whizfolders model of quality attributes in "Quality goals".

External qualities

primary sources = customers, users, and developers
inherent understanding risk level = medium

For a detailed description, see the link to the 3D Whizfolders model of quality attributes in "Quality goals".

Mixed qualities

primary sources = customers, users, and developers
inherent understanding risk level = high

For a detailed description, see the link to the 3D Whizfolders model of quality attributes in "Quality goals".

Functions

Domain functions

An acceptable set of domain functions must be congruent with the domain ontology and sufficient to achieve an application's goals (e.g., to sell merchandise). Goal support is expressed in terms of capabilities, functions, and attributes. This may result in a **hierarchy of capabilities** e.g., arranging a trip may entail arranging a flight, arranging a hotel, and arranging a car.

Interactive

Happy paths

primary sources = customers and users
inherent understanding risk level = medium

Unhappy paths

primary sources = developers
inherent understanding risk level = medium

Autonomous

Triggered by data changes or time.

primary sources = developers
inherent understanding risk level = low

System functions

Required functions provided by the application platform e.g., operating system. For example, backup and recovery functions.

primary sources = developers
inherent understanding risk level = low

Quality support functions

Functions supporting quality goals such as exception handlers, encryption routines, and safeguards.

A 3D Whizfolders model of quality attributes is freely downloadable from http://www.quality-aware.com/daves-q-a-stuff.php

External quality supports

primary sources = developers
inherent understanding risk level = medium

For details, see the link to the 3D Whizfolders model of quality attributes in "Quality support functions".

Mixed quality supports

primary sources = developers
inherent understanding risk level = high

For details, see the link to the 3D Whizfolders model of quality attributes in "Quality support functions".

Constraints

Required restrictions.

Technical

Design

Restrictions on design e.g., no single points of failure.

primary sources = developers
inherent understanding risk level = medium

Implementation

Restrictions on implementation e.g., programming languages and coding standards.

primary sources = quality management
inherent understanding risk level = low

Verification

Restrictions on verification activities e.g., test coverage requirements or specific targets and types of reviews.

primary sources = quality management
inherent understanding risk level = low

Deployment

Restrictions on deployment strategies

primary sources = developers
inherent understanding risk level = low

Societal

Societal restrictions such as laws, regulations, and policies.

primary sources = quality management
inherent understanding risk level = low

Project

Project restrictions such as budget, schedule, and staffing.

primary sources = project management
inherent understanding risk level = low

Supplier attributes

primary sources = quality management
inherent understanding risk level = low

Capabilities and health

Are supplier capabilities and health satisfactory?

a. Reputation and customer-base?
b. Corporate and product management?
c. Risk-management and quality-management culture?
d. Project management capabilities?
e. Relevant experience and capabilities of staff?
f. Capability maturity (at least SEI level 3 - Defined)?
g. Manageable competitive risk?
h. Financials?

Customer relationships

Is supplier easy to work with?

a. Is their understanding of domain, application, and requirements satisfactory?
b. Do they provide enough points of contact?
c. Are time zone and common language differences manageable?
d. Is communication open and supportive?
e. Are their responses and notices timely and helpful?
f. Do they help with drafting application requirements and acceptance criteria?
g. Are they sufficiently flexible?
h. Are they sufficiently dependable?
i. Are they proactive and innovative?
j. Is contracting with them cooperative or adversarial?

Change management

Is change management satisfactory?

a. Are their change and risk management procedures satisfactory?
b. Do they discuss changes in a timely manner?

Dependent application development activities

These are the development activities that are dependent on stakeholder understanding of the requirements. Lack of understanding or misunderstanding leads to defective results.

Design, architectural and detailed

Implementation

Verification

Segmented RD

Segmented requirements development (RD) recognizes that different RD strategies may be necessary for different parts of a system. Not only will system-level RD be different from component-level RD, but RD may be different for different aspects of a system. For example, in a safety-critical system, requirements for the safety-critical aspects may be different from those for the non-safety-critical aspects.

RD activities

Elicitation tactics and tools

Entails getting requirements-related information from customers, users, domain experts, and those experienced with similar applications. This means information about interactive happy paths, external qualities, and mixed qualities.

Other types of requirement information come from sources such as developers, quality managers, and project managers.

Observation

Document Analysis

Interface Analysis

Reverse Engineering

User responsibility identification

Use case development

Test design

Process modeling

Prototyping

Survey/Questionnaire

Individual & group discussions

Interview

Focus group

Workshop

About domain happy paths and quality attributes

Brainstorming

Specification tactics and tools

Entails documenting requirements-related information.

Because elicitation entails only some sources, specification may have a broader scope because it may entail information from any source.

for context elements

Ontologies or glossaries

Free-form natural language, lists, and tables

Domain facts

Specs may include Boolean expressions and/or arithmetic formulas

Charts and diagrams

Class and entity-relationship diagrams

for requirements information

Free-form natural language and lists

Specs may include Boolean expressions and/or arithmetic formulas

References

to increase precision

Controlled natural language

Formal specification language

Test cases

Conditions

Preconditions, postconditions, and invariant conditions.

Behavior rules

Action contracts

Mockups

3D quality models

to show relationships

Use cases

User stories or Scenarios

Acceptance test flows

Logic tables

State diagrams

Data flow diagrams

Prototypes

3D quality models

Analysis tactics and tools

Entails analyzing requirements-related specifications to assess their validity and task-adequacy.

Analysis tactics include:

- attribute analysis e.g., is requirement necessary?
- test design
- verification strategy review
- achievement strategy review
- skeptical inspection

Skeptical specification inspection

Requirements validation

Attribute analysis

For example, is the requirement necessary?

Test design

Behavior rule completeness analysis

Verification strategy inspection

Achievement strategy inspection

RD management

RD activity management

Promote learning

Assess requirements understanding

Plan requirements development

Facilitate communication

Manage requirements conflicts

Monitor status and results

RD Information management

There are many tools to support requirements information management.

Vision

Context

Vocabulary

Requirements

RD risk management

Risk management must be a central focus of project management. Requirements and their risks are part of this challenge.

Understanding and experience should be assessed to identify major hazards.

If understanding is just a little inadequate, limited learning can mitigate the problem. Otherwise, mitigation will need alternative actions.

Domain ontology inadequate understanding

Studying or developing a domain ontology or glossary may help.

In some cases, adequate domain understanding may require advanced study e.g., Masters in Organic Chemistry. In such cases, some developers must have domain expertise, since adequate understanding can't be acquired quickly enough.

Application inadequate understanding

Documenting and skeptically reviewing achievement and verification strategies.

Using high-impact inspections for skeptical peer reviews (reference 2.13 at www.understandingrequirements.com)

Requirements and their development inadequate understanding

Studying this ontology may help.

Developing metarequirements containing examples and references.

Promoting quality aware development. See http://www.quality-aware.com.

Specifying quality attribute requirements by tailoring a 3D quality model freely downloadable from http://www.quality-aware.com/daves-q-a-stuff.php.

Staffing a Quality & Productivity function (described in the Quality Goals chapter at http://www.quality-aware.com/daves-q-a-stuff.php).

Stakeholders

Stakeholders may be

- disruptive or lack understanding and may need to be replaced
- in conflict and may need arbitration

Insufficient resources

Critical stakeholders may be unavailable or under-involved.

Requirements development time may be insufficient.

These situations need to be communicated and addressed.