7. Requirements

22 September 2008

Bob Glushko

Plan for Today's Class

What are Requirements?

Context, "Point of View," and Requirements

Requirements in "Document-Intensive" Contexts

Methods for identifying, specifying, and communicating requirements

Business rules

Ravenflow demo

What Are Requirements?

When a problem or opportunity is defined and scoped, any constraints on possible solutions or goals that must be satisfied can be viewed as REQUIREMENTS: conditions that must be met for the solution to be acceptable to the stakeholders

Requirements are most often functional: descriptions of what the solution must do or must enable someone to do with it

Requirements can be expressed in terms of quality attributes (often called the "ilities" – usability, reliability, portability, interoperability, maintainability, etc.)

"Business Rule" is an up and coming buzzword in information systems that refers to a formal or refined statement of an requirement that is expressed in a technology-independent fashion

Requirements do not dictate how the solution is to be achieved – that is

"Context" Shapes Requirements

(1) Defining and scoping the problem and (2) Identifying the stakeholders are inextricably intertwined activities

We've previously discussed "the design context" and the "organizational context" in broad and somewhat coarse ways to help us do (1) and (2)

Today we'll refine and extend the idea of context to mean "the characteristics of the situation that define what is in or out of scope, inside or outside of the boundary in which our solution has to work"

Where this boundary falls depends on which stakeholder perspective(s) you emphasize

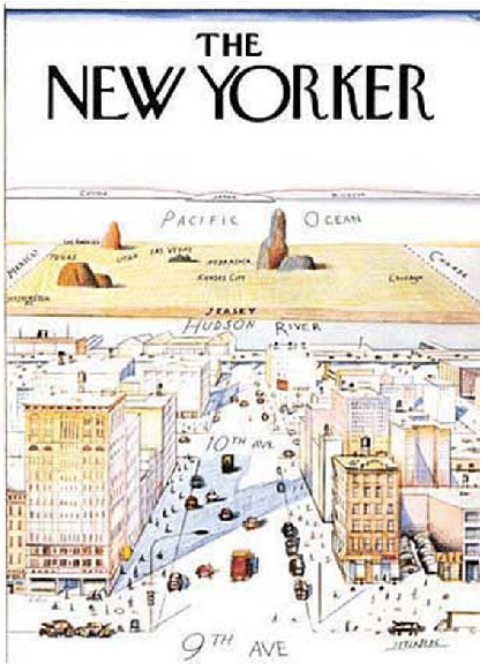
"Point of View" in a Design Context

In any service system design, we designate some component as the ultimate consumer or customer

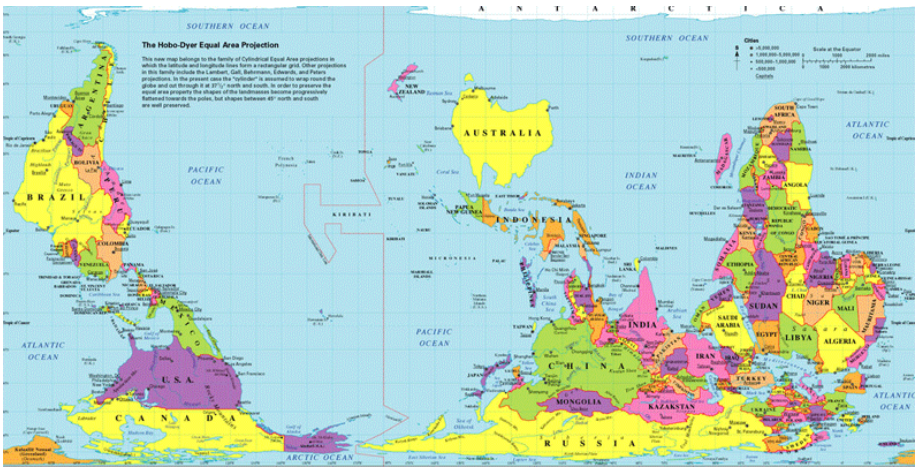
This often appears to be the end of a value chain or information flow, or where some typical class of "users" is found

But this point of view or perspective is often arbitrary, and there may be alternate POVs in the same service system

A New Yorker's Map



An Australian's Map



Who's the Service "Consumer" in a Teaching Hospital?

Front Stage and Back Stage Inversion: Cooking School, or Restaurant?

KITCHEN

Front Stage for the Cooks
Back Stage for the Customers

DINING ROOM

Front Stage for the Customers
Back Stage for the Cooks

Requirements in "Document Engineering" Contexts

Document Engineering focuses on the requirements for information and process models and for their computer-processable implementations

Many of the information requirements in document-intensive contexts are contained in existing documents

There is no sharp line dividing "requirements analysis," in which we get requirements from people, and "document analysis," in which we obtain them from documents.

Classifying Requirements in Document-Intensive Contexts

SOLUTION requirements – the functional, performance, quality attributes

INFORMATION or DATA requirements – what information is needed, what are its datatypes, possible values

DOCUMENT or STRUCTURE requirements – how is the information organized / assembled / packaged into sets of related information

PRESENTATION or SYNTACTIC requirements – how is the information presented or formatted or rendered – the physical or output model

PROCESSING and USAGE requirements – what relationships between documents have a business purpose

Setting the Context in "Document Engineering"

Any Document Engineering project involves some set of document types and information components that take part in some set of business processes for the benefit of some stakeholders

Because "no document (or process) is an island" there will always be some point at which the documents and processes you care about will intersect or overlap with some that that you don't care about

Some aspects of context are easy to specify because some business processes (and the documents that carry them out) are completely independent

But often there are interdependencies, and the challenge is to decide how many of them are important enough to consider and analyze

Context and Patterns

Much of what businesses do for themselves and with other businesses can be described using a small repertoire of supply chain or other business process / transaction patterns

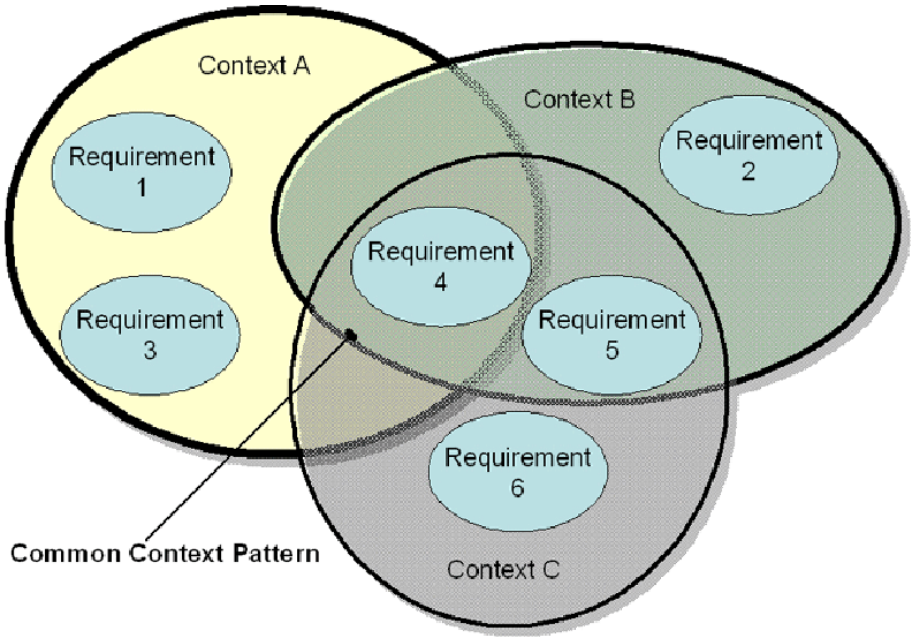
Each of these patterns implies a set of documents and some choreographies of document flows or exchanges

Selecting an appropriate pattern will help expose the information requirements, rules and constraints for our subsequent document analysis and design

Choosing a pattern suggests which document payloads we'll need to find or design and in which business processes we are likely to deploy them

How we describe context influences what patterns we identify and how we apply them

Patterns as Requirement Clusters; Contexts as "Contextualized" Patterns

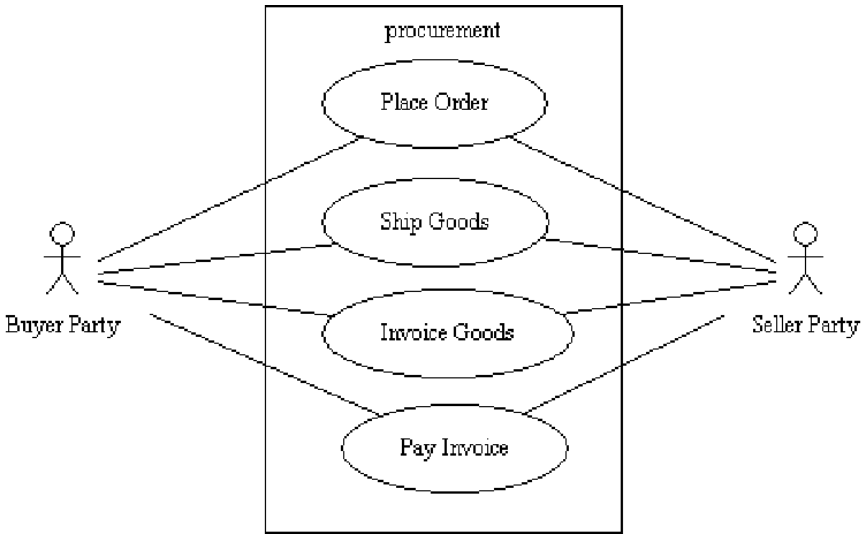


Business Process is the Most Important Context Dimension

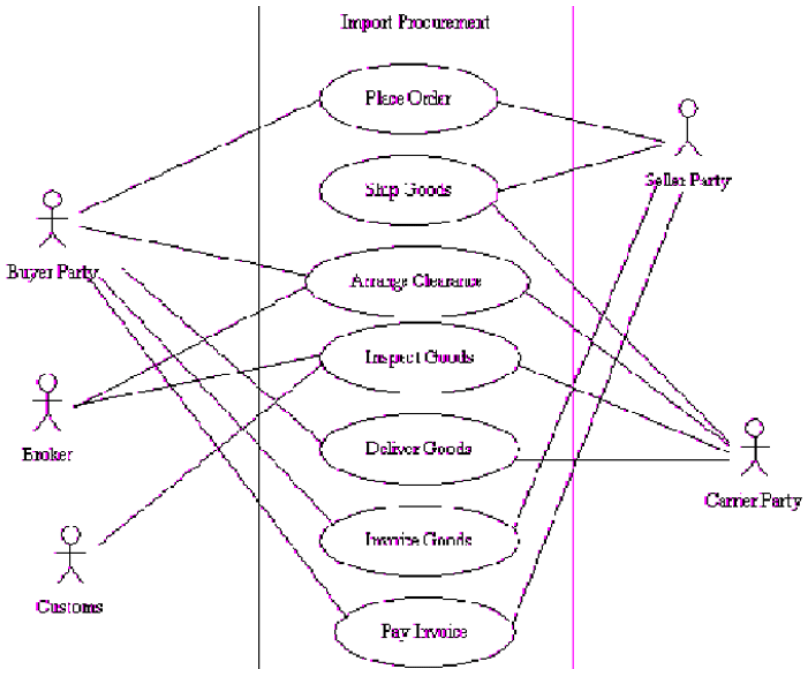
Business processes are the most important context dimensions because they highlight roles, perspectives, and strongly shape the information requirements

For example, contextualizing a simple procurement pattern to have the seller and buyer in different countries adds new processes and documents

Simple Procurement Pattern



Imported Goods Procurement Pattern



Other Context Dimensions

Product Classification

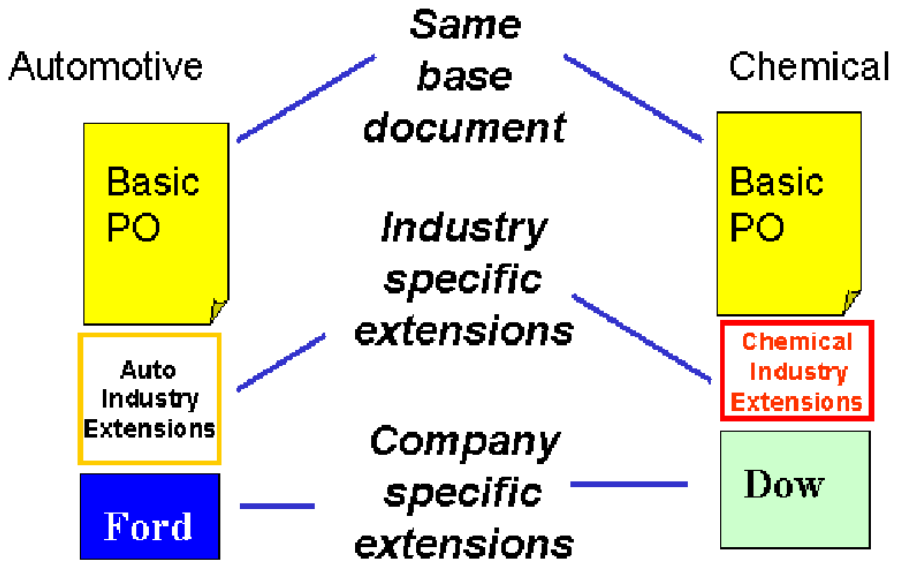
Industry Classification

Geopolitical Classification

Legal or Regulatory Jurisdiction

Business Process Role or Supporting Role

Representing Context in Document Architecture



Methods for Identifying Requirements

Studying people or phenomena in their natural surroundings as an anthropologist or ethnographer

Conducting face to face interviews or structured group sessions

Requirements "meta-models" or templates or lists of questions can yield more complete and consistently specified requirements

There are always lots of things people want to tell you if they trust you but which they will never write down

Responsibility vs Task-Centered User Classification

It is usually easy to identify people on the basis of their job titles or formal roles in the organization or environment that you are trying to understand – "system operator," "programmer," "shipping clerk," "truck driver," "professor," etc.

But this is less useful than classifying people on the basis of what they really do or what their actual responsibilities are, and the titles or formal roles may offer little help in discovering this

Job titles and formal organizational structure may not reflect what people actually do or what they might be doing better. Their job titles might make it difficult for them to tell you

Responsibility-Centered Classification and Requirements

Once you think in terms of responsibilities, you can define requirements in terms of satisfying them, which should produce better requirements

The tasks that people currently perform (or the documents they currently produce or use in performing them) might be constrained by their current systems or lack thereof. A new system (or new documents) might support better ways of meeting the responsibilities.

Paving the Cow Paths

In "document-intensive" processes (especially those where the documents are paper ones) it is seductively easy to define a requirement of "replacing paper forms with online ones" while otherwise leaving the basic processes the same

But this assumes that the product (or document) was designed to meet requirements in the first place and that these original requirements haven't changed

Most people concede that the "naturalist" perspective is needed if you're designing a new system or a new product (or a new document)

So taking time to identify requirements is desirable, if only to validate existing requirements, and it also ensures that important explicit and tacit knowledge isn't lost

Clarifying and Validating Requirements

If you did your job well in the "naturalist" phase, you can test your understanding of the requirements by conceptualizing alternative ways to satisfy them and trying to "sell" these to the people giving you the requirements

Testing your understanding means you won't end up "blaming the victim" for not expressing requirements correctly

Get people's impression of whether your approach would meet the need in an acceptable and usable way

Motivating "Rapid Requirements Definition"

Advances in software development tools and techniques have improved quality and timeliness in software-intensive design projects

But requirements definition is usually still slow, manual, and error-prone

The assumption that there is no hope for improving the requirements process has been a major motivation for "agile" software techniques

But even if agile techniques can get feedback on requirements that are visible in software artifacts, they can't identify other kinds of requirements from stakeholders other than the users of the software

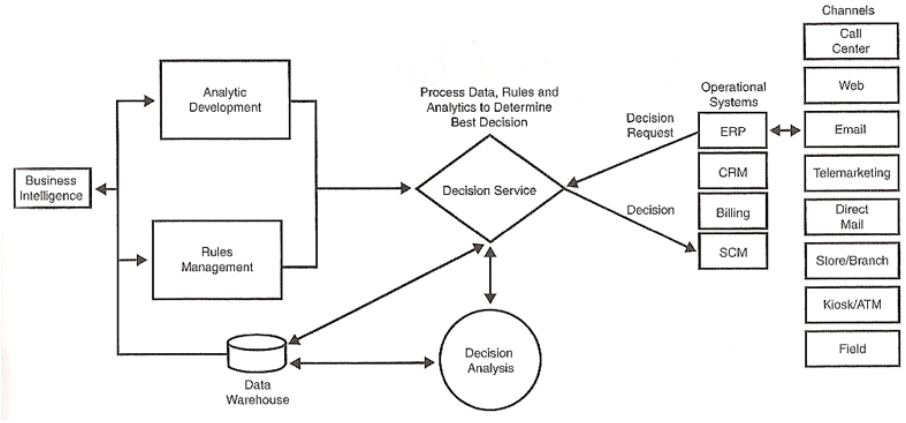
Specifying Requirements

Requirements can be quantitative or qualitative, but in all cases should be verifiable

PROCESS and DOCUMENT requirements are often closely related, and diagrams that depict their relationships – process / choreography / orchestration model that show document exchanges – are extremely helpful in ensuring that requirements are complete and consistent

Specifying requirements using business rules or using "simplified technical English" makes them easier to validate and communicate

"Decision Service" or "Business Rule" Architecture



From "Smart (Enough) Systems" by James Taylor & Neil Raden, p.145

Simplified Technical English

Simplified grammar - short sentences in active voice, simple verb tenses

Simplified logic - temporal or structural ordering of sentences with explicit "control flow"

Controlled vocabulary - approved and unapproved terms, used consistently

STE Example

BEFORE: Gain access to blade. After removing old blade, new blade may be fitted by proceeding in reverse order, using gloves to avoid injuries by teeth of blade. Before you attempt any of the above, the power should have been switched off.

AFTER: Make sure that the on/off switch is in the "off" position. Remove the blade cover from the machine. Warning: wear gloves when you touch the blade. Remove the used blade. Install the new blade. Install the blade cover.

Common Errors in Specifying Requirements

MISSING ACTOR: "If the order is incomplete, then..." -- Who determines that the order is incomplete?

FLOW BREAK, NULL "TRANSFER OBJECT" : "The clerk informs the manager" - How? By sending a document?

INCOMPLETE CONDITIONAL: "If x, then y" - What happens if not x?

"OTHERWISE" ORPHAN: "Otherwise" or "else" clauses that aren't preceded by and related to an "If" clause

INCONSISTENT NAMES: Are the "project lead," "project manager," and "lead engineer" the same actor?

Exercise: Identify Errors and Omissions in these Requirements

1. A requestor submits a Purchase Order Request to the purchasing department clerk
2. If the POR is for less than $5000, and if the POR has a valid budget number, then the clerk enters the POR.
3. The Purchasing System issues a PO and notifies the purchasing agent
4. Otherwise, the clerk rejects the POR

Communicating Requirements

For "big" designs, requirements are most commonly communicated in a large and formal textual document called a Business Requirements Document (BRD) or something similar like PRD (for "Product")

But the comprehensiveness and formality of such documents delays them and undermines their usability

The requirements analyst needs to take on the role of the "communication middleman" between the customer and the developer, and often between other types of stakeholders

So the analyst needs to be able to speak the "language" of each kind of stakeholder; "business rule" formats are designed to be the "hub language"

Categories for Business Rules

There are many different schemes for categorizing requirements or business rules.

Most approaches distinguish constraints on content that are represented in data models from constraints on behavior represented in process models

Other schemes distinguish single-item constraints that must always be true from those that reflect the relationship between two or more items or that can change according to a process context

Constraints can also be classified according to the manner in which they are represented and enforced in an implemented application

YART from Chapter 8 of Document Engineering

Rules that Apply to Conceptual Models

- Semantic
- Structural
- Usage

Rules that (Can Also) Apply to Physical Models

- Syntactic
- Processing
- Presentational

Rules that Apply to Instances or Implementations

- Content

Semantic Rules

Establish properties: "The Order Number is an identifier that is unique to the Buyer"

Express generalizations: "An Auto is a type of Vehicle"

Expose dependencies: "The Price of an Item can depend on the Buyer"

Structural Rules

Define co-occurrence or aggregation relationships between components

"Each Illustration must have a Caption"

"Every Order must have an Order Number, a Buyer and an Issue Date"

Usage Rules

Define policies or privileges governing access to information

Often based on organizational roles or responsibilities

"An Employee Salary can be viewed by a Manager but can only be changed by someone in the Human Resources Department"

Syntactic Rules

Concern the form in which models are encoded

Might specify a particular XML vocabulary

"All Purchase Orders must be encoded according to ANSI X12 850"

"All Purchase Orders must conform to the Universal Business Language"

Processing Rules

Define actions to be taken whenever a given condition occurs or set of information is encountered

Also called *Procedural* or *Behavioral* rules

"The Seller will respond to an Order with an Order Response"

"If the Item is Out of Stock, create a Back Order Request"

"If the Customer Account Balance is greater than $1000, do not allow any more purchases"

Presentation Rules

Govern the appearance or rendering of an information component

"The Item Description must be adjacent to the Product Image"

"The Title should be centered"

Content Rules

Establish constraints about the values of information components or dependencies among them

"The Total Value of an Order cannot exceed $100,000"

"The Currency Code should be expressed using ISO 4217 codes"

"The Shipping Address must be the same as the Billing Address"

"The Start Date must be earlier than the End Date"

"The Customer's Account Balance must be greater than 0"

Combined Rule Types

"If the Purchase Order Amount exceeds $10000, the Party must supply a Duns Number"

"If the Customer's Account Balance exceeds $1000, display it in Red"

"If the Customer's Account Balance exceeds $10000, send a Past Due Reminder by email, fax, and voice mail"

Readings for 24 September

Jan Kettinis, "Getting started with use case modeling"

Alistair Cockburn, ""“Structuring use cases with goals"

Larry Constantine & Lucy Lockwood, "Usage-centered engineering for web applications"