9

Domain Stakeholders

- The **prerequisite** for studying this chapter is that you have read Chaps. 1 and 8 of this volume.
- The **aims** are to introduce the concept of (domain) stakeholders, to distinguish between different categories of stakeholders, and to sketch a fairly advanced (also formalised) example of enterprise stakeholders.
- The **objective** is to ensure that you carefully consider and include the concerns of all relevant stakeholders when in future you are developing domain descriptions, requirements prescriptions and software designs.
- The **treatment** is from systematic to formal (sketches).

9.1 Introduction

At the very outset of any phase of development, whether the universe of discourse be some domain model development, requirements development or software design, it is important to identify all possibly relevant stakeholders. Throughout the development phase it is then important to ensure that each stakeholder (group) is properly "taken care of", i.e., their concerns are properly modelled.

9.2 Stakeholders

Characterisation. By a domain *stakeholder* we shall understand a person, or a group of persons, "united" somehow in their common interest in, or dependency on the domain; or an institution, an enterprise, or a group of such, (again) characterised (and, again, loosely) by their common interest in, or dependency on the domain. ■

Obviously we could "equate" institutions and enterprises with groups of one or more persons. For pragmatic reasons of identification (i.e., "discovery") it is, in cases, sometimes easier, we believe, to think of institutions and enterprises.

9.2.1 General Application Stakeholders

Characterisation. By *general application domain stakeholders* we understand stakeholders whose primary interest is neither the projects which develop software (from domains, via requirements to software design), nor the products evolving from such projects. Instead we mean stakeholders from typically non-IT business areas. ▪

Thus general application domain stakeholders are typically those whom we can characterise as from such domains as: transportation, manufacturing, mining, financial industries, public government, the service sector, etc.

Example 9.1 *Railway Train System Stakeholders:* When modelling, i.e., describing, the domain of railways, one may be well advised in considering the following stakeholder groups — listed in an order that may reflect the view of the first group: (i) owners (e.g., stockholders or a government), (ii) management (consisting of (ii.1) executive management, (ii.2) mid-level management, (ii.3) operational ("floor") management (i.e., "white collar workers"), etc.), (iii) railway staff at large ("people on the floor" other than "floor" management, i.e., "blue collar" workers — and possibly arranged into several stakeholder groups including families), (iv) customers ((iv.1) passengers and (vi.2) freightors (people etc., sending and receiving freight)), (v) users ((v.1) people coming to send off or receive passengers, and (v.1) people coming to send off or receive freight), (vi) agents ((vi.1) travel bureaus, and (vi.2) logistics firms), (vii) railway infrastructure companies,[1] (viii) suppliers ((viii.1) of day-to-day materiel (consumables), (viii.2) of new railway infrastructure components (i.e., lines, tracks, signals, etc.), and (viii.3) of information technology and software), (ix) railway regulatory agency or agencies, (x) politicians "at large", and (xi) the general public, "at large". ▪

The above example is typical of the kind of rough sketch, or even narrative, documentation text that the software developer has to produce in the course of describing a domain. But the above list is merely indicative, not final. It is also given here to "augment" our characterisation of what is meant by the stakeholder concept. Thus you can take this listing as a cue to construct similar stakeholder listings for other domains.

9.2.2 Software Development Stakeholders

One can identify two extremes of software (SW) developments: turnkey software and commercial off-the-shelf software (COTS).

[1] So we are considering a train service operator, in contrast to those who own (and operate) the rail net. In many countries these are two distinct groups of enterprises.

Turnkey Software Development Stakeholders

Characterisation. By *turnkey software* we understand software that is developed — usually from "almost scratch" — in very specific response to a specific client/developer contract. ▪

Characterisation. By a *turnkey software development stakeholder* we thus understand a stakeholder from the software developer or from that client domain. ▪

Stakeholders from this "extrovert" domain are (thus) typically: (i) The client (i.1) contract management, (i.2) client users and (i.3) customers, affected by the contracted software; and (ii) the software house (ii.1) contract management, (ii.2) software engineers and (ii.3) supporting technicians.

Commercial Off-the-Shelf (COTS) SW Development Stakeholders

By COTS we mean generic kinds of software, i.e., software whose functionality is as much, or more, decided upon by the makers of the software than by the customers and users of the software; software which is expected to cover, or actually covers the needs of many clients, and which the maker thus expects to sell in dozens, hundreds or thousands of copies.

Characterisation. *COTS Stakeholder:* By *COTS stakeholders* we (thus) typically mean: people from software houses: (i–ii) software house owners and management (at least two groups), (iii–v) marketing, sales and service departments (three groups), (vi) the programmers, i.e., the software engineers, (vii) distributors of the software, and (viii) other software houses which base tailor-made software development on COTSs; as well as people from the application domains for which the software house makes these products: (ix) customers (clients) and (x) users. ▪

9.2.3 Purpose of Listing Stakeholders

Lest we forget it, let us remind ourselves why we wish to systematically record all possibly relevant stakeholder groups: It is so that we can systematically and "near exhaustively" consider all relevant stakeholder groups, when we now go on to ascertain their view of, their perspective of, the universe of discourse — here the domain.

9.3 Stakeholder Perspectives

Characterisation. By a *stakeholder perspective* we understand the, or an, understanding of the domain shared by the specifically identified stakeholder group — a view that may differ from one stakeholder group to another stakeholder group of the same domain. ▪

For each stakeholder group we have to investigate (elicit, acquire, and analyse) its perspective with respect to each of the possible domain attributes, as covered next, and each of the possible domain facets, as covered thereafter. With respect to stakeholder perspectives we may be prepared to observe that one and the same phenomenon may be considered by two different groups to possess not quite commensurate attributes, and not quite commensurate facets. And thus two or more such group perspectives can give rise to inconsistent, and/or conflicting overall views on domain attributes and facets. We shall return to the above issues when we later treat the methodological concerns of domain acquisition and validation.

9.3.1 Perspectives of General Applications

The stakeholder perspectives for general application domains are generally of several concerns:

(i–ii) *Client executive and other upper-level management* expects computing systems to improve their company's competitiveness, financial position, etc. These are issues that are very hard to formulate, let alone formalise. Under informative documents we list part of these concerns under the headings *assumptions and dependencies* and *implicit/derivative goals*. For more on this we refer to Sect. 2.4.6.

(iii) *Tactical and operational management* usually have perspectives that pertain to management and organisational issues.

(iv) *Nonmanagement staff* usually have perspectives that pertain to their daily work and to its interface with customers.

(v) All of the above stakeholder groups have perspectives that primarily focus on their shared domain: the general application area.

(vi) This is in contrast to the perspectives of stakeholders of the software house, the developer with whom the client contracts.

(vii) Besides wishing to secure, in their perspectives, the professional integrity of their company, the *software house developer* perspectives include those of satisfying the client.

Example 9.2 *Resource Management:* We now present a rather lengthy example that illustrates the interface between a number of stakeholder perspectives. The stakeholders are (simplifying): an enterprise's top level, executive management (who plan, take and follow up on strategic decisions), its line management (who plan, take and follow up on tactical decisions), its operations management (who plan, take and follow up on operational decisions) and the enterprise "workers" (who carry out decisions through tasks). The management groups have the following kinds of functions. Strategic management has to do with upgrading or downsizing, i.e., converting an enterprise's resources from one form to another — making sure that resources are available for tactical management. Tactical management has to do with temporal, typically medium- to long-term scheduling and spatially allocating these resources, in preparation for operations management. Operations management plans final (usually short-term) scheduling and allocation of (resource-consuming) tasks, in preparation for actual enterprise ("floor") operations.

After some analysis we arrive at the following: Let R, Rn, L, T, E and A stand for resources, resource names, spatial locations, times, enterprises (with their estimates, service and/or production plans, orders on hand, etc.), respectively tasks (actions). SR, TR and OR stand for strategic, tactical and operational resource views, respectively. SR expresses (temporal) schedules: which sets of resources are either bound or free in which (pragmatically speaking: overall, i.e., "larger") time intervals. TR expresses temporal and spatial allocations of sets of resources, in certain (pragmatically speaking: model finer-grained, i.e., "smaller") time intervals, and to certain locations. OR expresses that certain actions, A, are to be, or are being applied to (parameter-named) resources in certain time intervals.

Formal Presentation: Resource Management

type R, Rn, L, T, E
  RS = R-**set**
  SR = $(T \times T) \xrightarrow{m} RS$, SRS = SR-**infset**
  TR = $(T \times T) \xrightarrow{m} RS \xrightarrow{m} L$, TRS = TR-**set**
  OR = $(T \times T) \xrightarrow{m} RS \xrightarrow{m} A$
  A = $(Rn \xrightarrow{m} RS) \overset{\sim}{\rightarrow} (Rn \xrightarrow{m} RS)$
value
  obs_Rn: $R \rightarrow Rn$
  srm: $RS \rightarrow E \times E \overset{\sim}{\rightarrow} E \times (SRS \times SR)$
  trm: $SR \rightarrow E \times E \overset{\sim}{\rightarrow} E \times (TRS \times TR)$
  orm: $TR \rightarrow E \times E \overset{\sim}{\rightarrow} E \times OR$
  p: $RS \times E \rightarrow$ **Bool**
  ope: $OR \rightarrow TR \rightarrow SR \rightarrow (E \times E \times E \times E) \rightarrow E \times RS$

The partial, including loosely specified, and in cases nondeterministic functions srm, trm and orm stand for strategic, tactical, respectively operations resource management. p is a predicate which determines whether the enterprise can continue to operate (with its state and in its environment, e), or not. To keep our model small we have had to resort to a "trick": putting all the facts knowable and needed in order for management to function adequately into E. Besides the enterprise itself, E also models its environment: that part of the world which affects the enterprise.

There are, accordingly, the following management functions:

Strategic resource management: srm(rs)(e,e''''). Let us call the result (e',(srs,sr)) [see "definition" of the enterprise "function" below]. srm proceeds on the basis of the enterprise: as it is now (e), and as one would like it to become (e''''), as well as its current resources (rs). srm "ideally estimates" all possible strategic resource acquisitions (upgrading) and/or downsizings (divestments) (srs). And srm selects one desirable strategic resource schedule (sr). The “estimation” is heuristic. Too little is normally known to compute sr algorithmically. One can, however, based on careful analysis of srm’s pre/-postconditions, usually provide some form of computerised decision support for strategic management.

Tactical resource management: trm(sr)(e,e′′′′). Let us call the result (e′′,(trs,tr)). trm proceeds on the basis of the enterprise: as it is now (e), and as one would like it to become (e′′′′), as well as one chosen strategic resource view (sr). trm “ideally calculates” all possible tactical resource possibilities (trs). And trm selects one desirable tactical resource schedule and allocation (tr). Again, trm cannot be fully algorithmitised. But some combinations of partial answer computations and decision support can be provided.

Operations resource management: orm(tr)(e,e′′′′). Let us call the result (e′′′,or), orm proceeds on the basis of the enterprise: as it is now (e), and as one would like it to become (e′′′′), as well as one chosen tactical resource view (tr). And orm effectively decides on one operations resource view (or). Typically orm can be algorithmitised — applying standard operations research techniques.

Actual enterprise operation: ope, enables, but does not guarantee, some “common” view of the enterprise. ope depends on the views of the enterprise, its context, state and environment, e, as “passed down” by management; and ope applies, according to prescriptions kept in the enterprise context and state, actions, a, to named (rn:Rn) sets of resources.

The above account is, obviously, rather idealised, but, we hope it is indicative of what is going on. Relating the above schematic example to, for example, the railway domain we may suggest: Resources R include access to (not necessarily ownership of) the rail net, rights to rent passenger train carriages and locomotives, staff, monies, etc. Strategic resources is, for example, about needing additional or changed rail net access rights, needing further or different kinds of train sets, etc. Strategic resource management, srm, typically brings many operators together, negotiating with rail infrastructure owners about access rights and with train set leasing (and lease finance) companies for rental of train sets, etc. srs:SRS designates all possible outcomes of a company’s own strategic planning; sr:SR designates a negotiated solution. Tactical resources is, for example, now about the rostering of train staff (crew allocation), allocation of train sets to maintenance locations, etc. Tactical resource management, trm, typically involves negotiation with trade unions, with maintenance units, etc. trs:TRS designates all possible outcomes of a company’s own tactical planning (its negotiating options); tr:TR designates a negotiated solution.

To give a further abstraction of the “life cycle” of the enterprise, we idealise it, as now shown:

value

```
enterprise: RS $\overset{\sim}{\to}$ E $\overset{\sim}{\to}$ **Unit**
enterprise(rs)(e) $\equiv$
    **if** p(rs)(e) **then**
        **let** (e′,(srs,sr)) = srm(rs)(e,e″″),
            (e″,(trs,tr)) = trm(sr)(e,e″″),
            (e‴,or) = orm(tr)(e,e″″),
            (e″″,rs′) = ope(or)(tr)(sr)(e,e′,e″,e‴) **in**
        **let** e‴″′:E • p′(e″″,e‴″′) **in**
        enterprise(rs′)(e‴″′) **end end**
    **else stop end**

p′: E × E → **Bool**
```

The enterprise reinvocation argument, rs′, a result of operations, is intended to reflect the use of strategically, tactically and operationally acquired, spatially and task allocated and scheduled resources, including partial consumption, "wear and tear", loss, replacements, etc.

The **let** e‴″′:E • p′(e″″,e‴″′) **in** ... shall model a changing environment.

There were two forms of recursion at play here: The simple tail-recursion (i.e., the recursive invocation of enterprise), and the recursive "build-up" of the enterprise state e″″. The former is trivial. The latter is the interesting one: Solution, by iteration towards some acceptable, not necessarily minimal fix-point, "mimics" the way the three levels of management and the "floor" operations change that state and "pass it around, up and down" the management hierarchy. The operate function "unifies" the views that different management levels have of the enterprise, and influences their decision making. Dependence on E also models potential interaction between enterprise management and, conceivably, all other stakeholders.

We remind the reader that — in the previous example — we are "only" modelling the *domain!* That model is, obviously, sketchy. But we believe it portrays important facets of domain modelling and stakeholder perspectives. The stakeholders were, to repeat: strategy ("executive") management (srm, p), tactical ("line") management (trm), operations ("floor") management (orm) and the workers (ope). The perspective being modelled focused on two aspects: their individual jobs, as "modelled" by the "functions" (srm, p, trm, orm, ope), and their interactions, as "modelled" by the passing around of arguments (e, e′, e″, e‴, e″″). The **let** e‴″′:E • p′(e″″,e‴″′) **in** ..., which "models" the changing environment, thus summarises the perspectives of "all other" stakeholders!

We are modelling a domain with all its imperfections: We are not specifying anything algorithmically; all functions are rather loosely, hence partially defined; in fact only their signature is given. This means that we model well managed as well as badly, sloppily or disastrously managed enterprises. We can, of course, define a great number of predicates on the enterprise state and its environment (e:E), and we can partially characterise intrinsics — facts that must always be true of an enterprise, no matter how.

If we "programme-specified" the enterprise then we would not be modelling the domain of enterprises, but a specifically "business process engineered" enterprise. Or we would be into requirements engineering — we claim. ■

9.3.2 Perspectives of Software Development

If the application domain is that of software development itself then the domain stakeholders are primarily the software house owners and upper management, the software engineers and their immediate managers, the technicians who support the work of the software engineers, and the suppliers of technology (hardware and software) that support the work of management, software engineers and technicians. This is true whether the software development is either just domain engineering, or just requirements engineering, or just software design, or the first two, the last two or all three of the above. We stress the precondition: "if the application domain is that of software development itself". Or, put it differently, the subject domain of these volumes is software development itself.

9.4 Discussion: Stakeholders and Their Perspectives

9.4.1 General

We refer to Chap. 18 for a treatment of requirements stakeholders. This chapter has discussed the concept of stakeholders. In subsequent chapters we shall take up the thread and, occasionally, indicate where we differentiate, in our descriptions, etc., between perspectives of different stakeholders. In Chap. 10 this will not be an issue, but in Chap. 11's treatment of business processes and management and organisation we may occasionally refer to the need for special descriptions of stakeholder perspective.

9.4.2 Principles, Techniques and Tools

Principle. *Domain Stakeholder:* At the very outset of a development project identify all possible and potential domain stakeholders. It is better to include too many, than forget some who can later create a nuisance, or more, when rightfully intervening. Be prepared, throughout a project, to revise the list of domain stakeholders. ■

Principle. *Domain Stakeholder Perspective:* At the very outset of a development project define, together with designated domain stakeholders, their roles, their "jurisdictions" and their "rights and duties". Be prepared, throughout a project, to revise the roles of domain stakeholders. ■

Techniques. *Domain Stakeholder Liaison:* (i) Maintain, openly inspectable, lists of all contemplated, respectively of all actual domain stakeholders. (ii) Liaise regularly with all actual domain stakeholders. (iii) Inform all other (contemplated) domain stakeholders of "what's going on". (iv) Write down in clear (natural, yet legally binding) language the role of each actual stakeholder. (v) Maintain a dossier of all communications with all domain stakeholders. Typically such communications deal, as we shall see later, with: role assignments, acquisition and validation. ▪

Tools. *Domain Stakeholder Liaison:* The tools mentioned under information documents (Sect. 2.4.10) apply equally well here. ▪

9.5 Exercises

9.5.1 Preamble

The first 4 exercises (9.1–9.4) of this chapter are *closed book* exercises. That means that you are to try write down a few lines of your solution before you check with the appropriate section for our answer to the questions.

9.5.2 Assignments

Exercise 9.1 *Domain Stakeholder.* This is a "repeat" question (see Exercise 8.8): Without consulting chapter texts in these volumes, try to characterise, in a few lines, how this chapter defines the concept of a domain stakeholder.

Exercise 9.2 *Domain Stakeholder Perspective.* This is a "repeat" question (see Exercise 8.9): Without consulting chapter texts in these volumes, try to characterise, in a few lines, how this chapter defines the concept of domain stakeholder perspective.

Exercise 9.3 *General Application Versus Software Development Stakeholders.* Without consulting chapter texts in these volumes, try to enumerate, in three or so lines, how this chapter perceives of a spectrum from general application stakeholders to software development stakeholders.

Exercise 9.4 *General Application Versus Software Development Stakeholder Perspectives.* Given your answer to the previous exercise (Exercise 9.3 above), augment the answer by providing each entry in your enumerated list with a brief characterisation of corresponding perspectives.

Exercise 9.5 *Domain-Specific Stakeholders.* For the fixed topic, selected by you, present as exhaustive a list of stakeholders as you think is relevant for your domain modelling.

Exercise 9.6 *Domain-Specific Stakeholder Perspectives.* Given your answer to the previous exercise (Exercise 9.5 above), augment the answer by providing for each entry in your enumerated list a brief characterisation of corresponding domain-specific perspectives.

9.5.3 Postlude

Exercises similar to the above will be reposed in Sect. 18.5.2.