Documenting the ICT Architecture of TSI

R. J. Wieringa[1] H. Blanken[1] M. Fokkinga[1]

December 12, 2001

[1]Department of Computer Science, University of Twente, P.O. Box 217, 9700 AE Enschede, the Netherlands, (roelw|blanken|fokkinga)@cs.utwente.nl

Contents

Chapter 1

Introduction

In this report, we discuss the business- and software architecture of Travel Service International (TSI). We use this as a case study to validate our approach to ICT architecture. The techniques discussed in this report are explained at length in a textbook [1].

1.1 Layered Architecture

Our approach is based upon a layered view on architecture, in which we distinguish six layers (figure 1.1).

- The business environment consists of the value chain in which the business operates. This includes clients, suppliers, competitors, government bodies, distribution- and communication channels.
- The business is an organization of people and machines with a common purpose to deliver a product or service to a market. The common purpose is stated in a shared mission statement. The delivery of products or services is summarized in a list of external functions.
- Business processes are
  - operational processes that respond to external and temporal events to deliver products and services,
  - supporting processes, and
  - strategic and tactical management processes.
- Application software supports or fully performs part of the operational and other business processes.
- The software platform is the collection of standard general-purpose software needed to run the application software. It is also called "implementation platform". It ranges from operating systems, middleware, network software to database management software.
- The network hardware consists of the physical resources that run the software platform and the application software. "Physical" means "having a size and weight". The network consists of boxes that contain metal, plastic and silicium, glass screens, copper wires and other physical entities.

Each line between two layers generally represents a many-many mapping between the two adjacent layers.

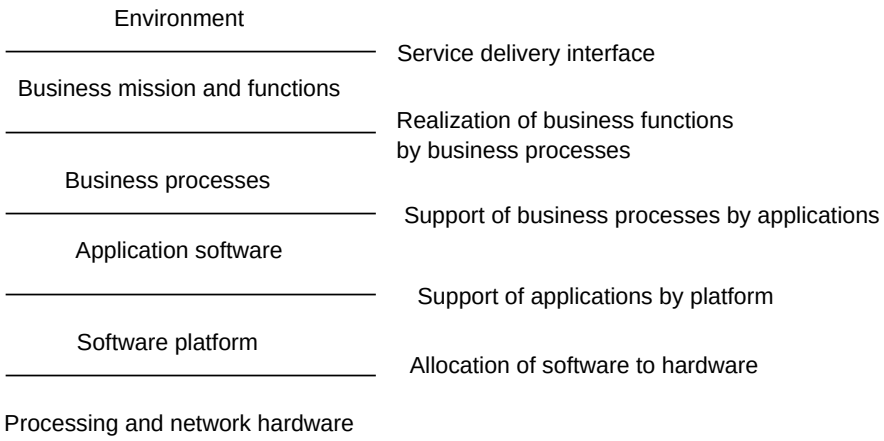

Figure 1.1: Layered view. Components are outsourced. Each line is a many-many mapping.

- The service delivery interface maps events in the environment to services to be delivered by the business in response to those events.
- Functions are realized by business process. One function may need several processes and one process may support several functions.
- Each application supports several business processes and each process can be supported by several applications.
- Each application uses several software platform entities and each platform entity supports several applications.
- Each software system (application or platform entity) runs on one or more network nodes and each such node runs several software systems.

The relationship between the levels is that lower levels provide services to higher level. Services at one level may be used by several entities at the higher levels.

1.2 Method

Our methodological approach to architectures is both top-down and middle-out (figure 1.2). It is top-down in the sense that we start with modeling the environment and proceed to lower layers. It is also top-down in that we start with descriptions that keep a high level of abstraction and proceed to descriptions that have a lower level of abstraction (which is a high level of detail). The trick to make it a middle-out process is to see that refinement is independent from layering. We can describe any layer at any level of detail. When we describe lower service levels, we generally require more detail at this level and at all higher levels. So our initial environment description is quite abstract, but when we describe the application software, we need a more detailed description of the business environment and of the business processes.

When we add detail, we must choose the *aspect* about which we want to add detail. For example, when we add detail about the environment, we can add technical details (e.g. which communication technology is used), or legal details (e.g. which laws are applicable), or financial detail (where do cash flows come from and go to) etc.

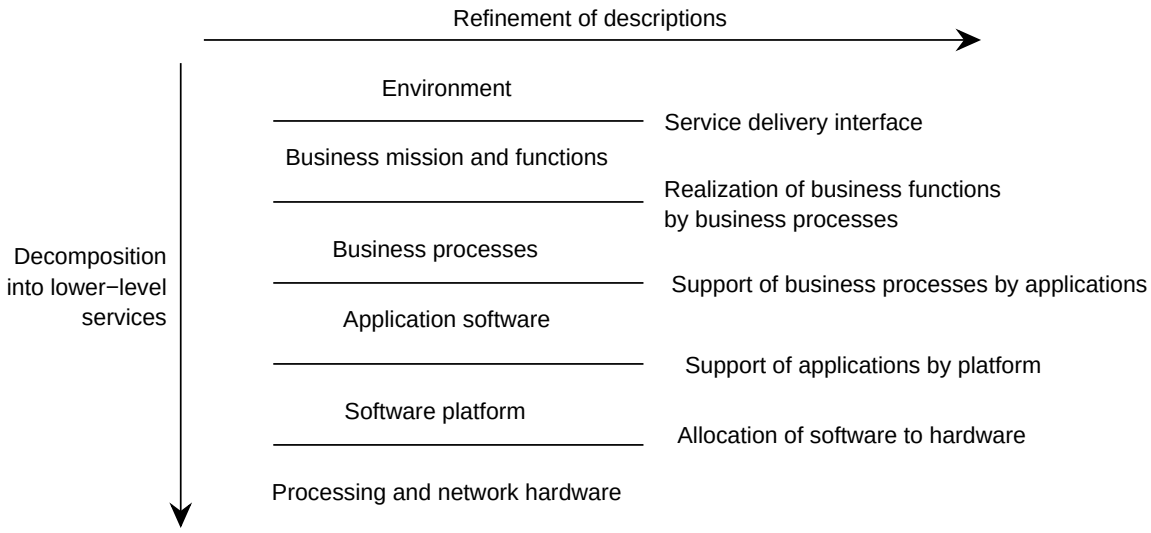

Figure 1.2: Our methodological approach. Going to a lower level may require adding more detail to higher-level descriptions.

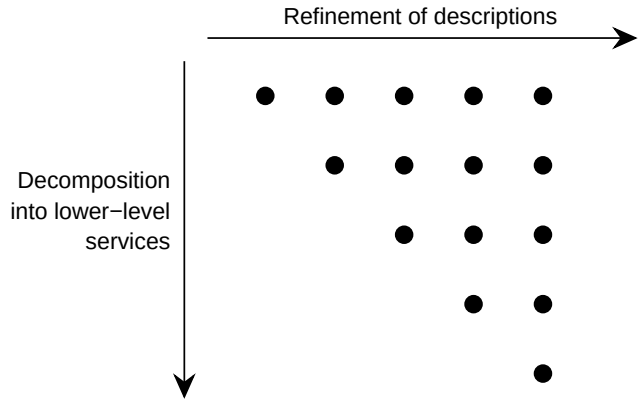

Figure 1.3: Description of lower layers requires adding more detail to descriptions of higher layers.

And the choice of aspect that we choose to detail depends upon the kind of lower-level service that we wish to describe. For example, if we describe the services of application software, we will need other kinds of detail about the business environment and business processes than we need when we describe the kind of furniture to be put in the building, or when we describe the social processes in the business. To describe application software, we need a model of the entities about which the software must store information and a model of the processes that will provide the software with data about these entities. When we describe the desired furniture, we need information about the image that the business wants to project to its customers, and about the ergonomic properties required by the people who perform the business processes.

Figure 1.3 illustrates this. We start at the upper left corner with a simple description of the environment. When we describe the business mission and function, we may want to add more detail to the environment description, such as the communication channels through which the business communicates with other parties. Descending to the business processes, we may want to add yet more detail to the environment model, for example by adding a list of relevant external events to which the business should respond. Etcetera.

In the appendix, we summarize the models we think are relevant for each layer and level of refinement. One description we mention at the outset: The dictionary. From our very first environment model, we will start a dictionary of important terms, that will be extended as we add more detail. The dictionary supports the primary function of the models: Communication between people.

1.3 Design Charter and Environment

A second elaboration of the simple top-down approach is that we do not start from scratch. In a greenfield approach, the environment would be given and the business is to be designed. As a matter of fact, the environment and many parts of the business are given and some parts of the business may be redesigned.

For example, we may have a charter for changing only the grey rectangle in figure 1.4 but must leave the rest of the business unchanged. These unchanging entities and processes are the *environment* of our design activity. It is important to see that the design environment is not the same as the business environment. The design environment exists at every level of our hierarchy.

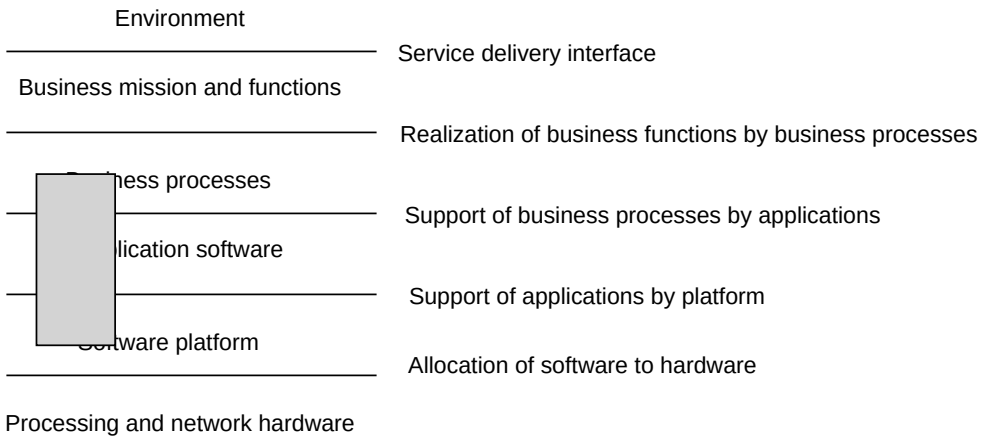

Figure 1.4: Design environment and charter for change.

The subject of our design decisions is bound by our *charter* to change part of the world. We may have a charter to change business processes but leave application software unchanged, or to change application software but leave business processes unchanged, to change the software platform and leave everything else unchanged, etc. Figure 1.4 gives simplified picture, for a charter for change may contain entities scattered over many layers and the design environment may be scattered over the very same layers.

Chapter 2

The Environment of TSI

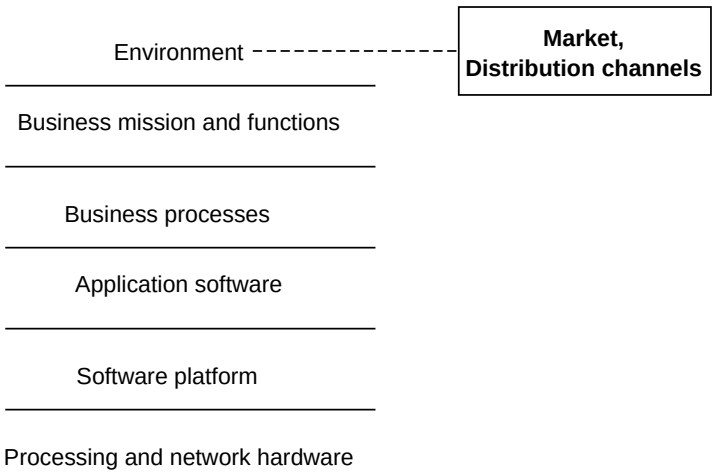

Figure 2.1: Focusing on the environment. Our first environment description focuses on market and distribution channels.

Figure 2.2 shows the market in which TSI operates. Figure 2.3 shows this in a rich picture. ()A rich picture is a cartoon-like figure that represents a domain by a number of icons, with totally informal and possibly ambiguous and ad hoc semantics, used to guide a discussion about the domain.) TSI communicates with their customers, travelers and home owners by means of email, web pages, fax and telephone.

Key terms used in this report are defined in figure 2.4.

The market of TSI is application-service provision for on-line rent of holiday homes.

- Clients are private individuals seeking to rent holiday homes.
- Suppliers are private individuals seeking to rent out their holiday home, VVV (the Dutch tourism service) and NBT (Dutch Bureau of travel agents).
- There are currently no competitors. Companies like KPN could develop subsidiaries that would compete with TSI.

Current threats are the dip in the travel industry and in Internet business. The largest opportunity is that TSI has the software to help other companies take the next large step in e-business.

Figure 2.2: Market of TSI.

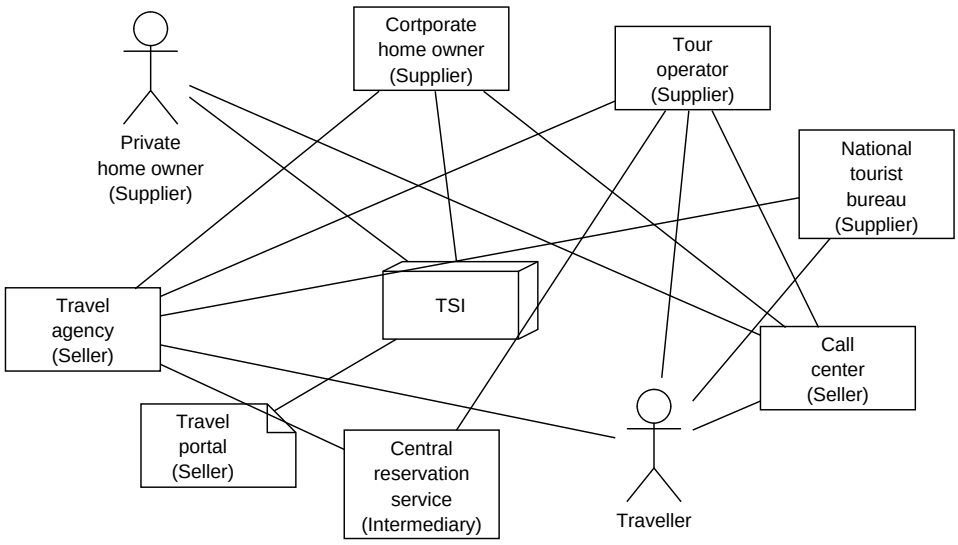

Figure 2.3: Rich picture of the market in which TSI operates.

- **Customers** are sellers or suppliers.
- **Sellers** of TSI are private or corporate individuals seeking to rent out holiday homes. Examples are travel agencies, call centers, travel portals. One seller can act on behalf of many home owners.
- **Suppliers** are private or corporate entities that own homes that they offer for rent by travelers, either directly or through sellers. Examples are private home owners, tour operators and national tourist bureaus.
- **Travelers** are private individuals seeking to rent holiday homes.

Figure 2.4: Dictionary.

Chapter 3

Mission and Services of TSI

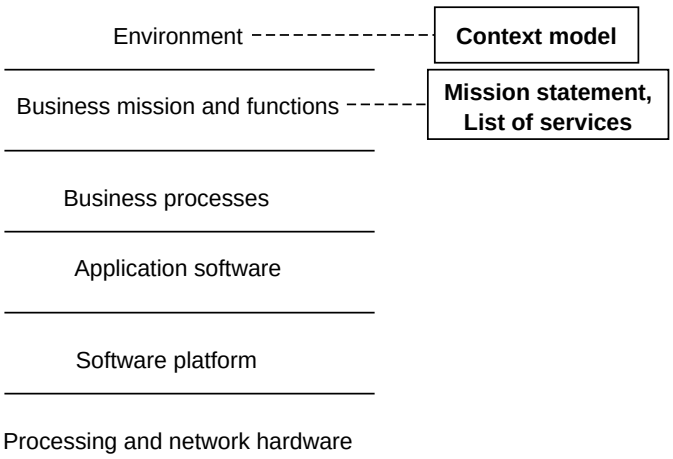

Figure 3.1: Focusing on the business services. We add a context model of the environment.

The business mission and services of TIS are listed in figure 3.2. The service decomposition tree is shown in figure 3.3.

Figure 3.4 shows the context in which TSI is working.

Sellers are private or corporate individuals seeking to rent out holiday homes. Examples are travel agencies, call centers, and travel portals. *Suppliers* are private or corporate entities that own homes that they offer for rent by travelers, either directly or through sellers. Examples are private home owners, tour operators and national tourist bureaus. The context diagram shows the communication channels in use. It shows that suppliers can rent out homes directly to travelers, or can do that through a seller, or through TSI. A seller in turn can rent out homes directly to travelers or through TSI.

Observe that the description of the services gives us the first outline of the subject domain. TSI talks with its external entities about holiday homes and reservations. So these entities will be in its subject domain and we can expect that TSI must store information about them.

Mission:

- to host databases and transactional web pages on behalf of customers, so that travelers can search for houses and perform on-line booking.

Services provided:

- Customer services:
  - Initialize web site and database
  - Update web site and database
  - Provide access and sales statistics
- Traveler services:
  - Provide information about holiday homes.
  - Offer reservation capability.
  - Offer payment capability.
  - Offer feedback capability.

The major services acquired from other parties are:

- Payment handling
- Authentication services

Figure 3.2: Mission and external functions (services) of TSI.

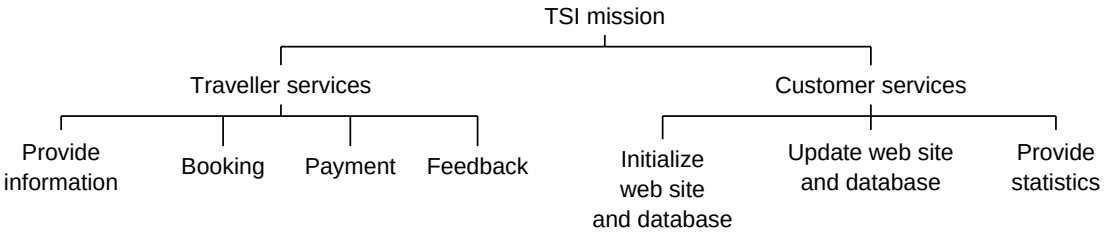

Figure 3.3: Service decomposition tree.

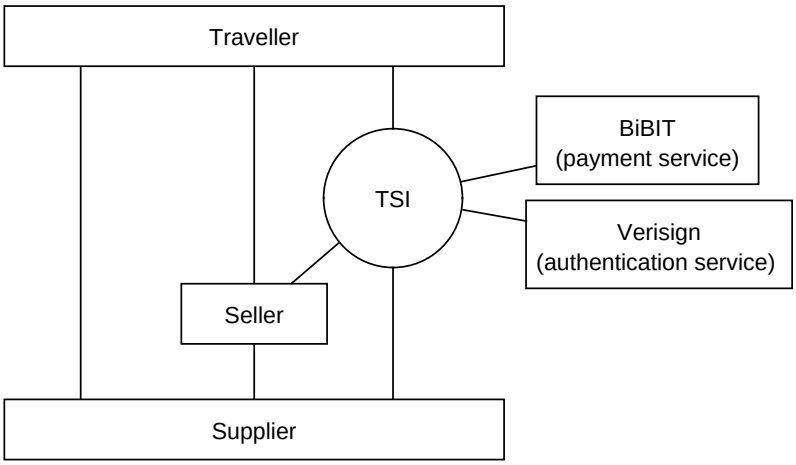

Figure 3.4: The business context model

Chapter 4

Business Processes

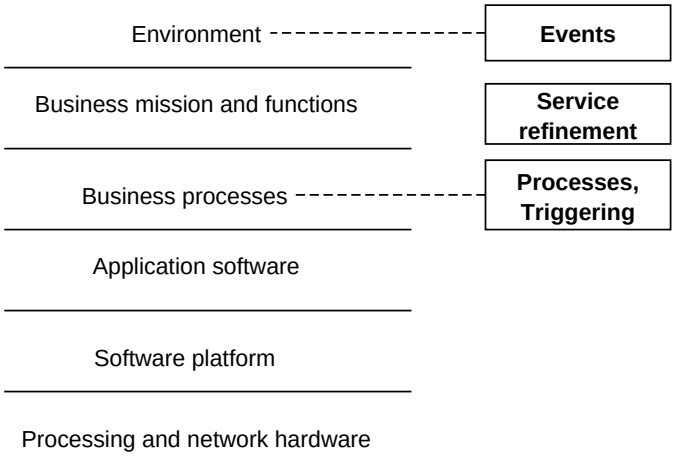

Figure 4.1: Focusing on the business processes. We add a model of events and show how these trigger business processes, and which services are delivered that way.

Each operational business process handles events. It responds to them by updating the state of the business and by sending a response to the environment. This happens in the following pattern (figure 4.2): An *external event* occurs in the environment and leads, through some communication channel, to a some *stimulus* at the interface of the business. This leads to a business process that produces a *response*, which through some communication channel, leads to a desired action in the

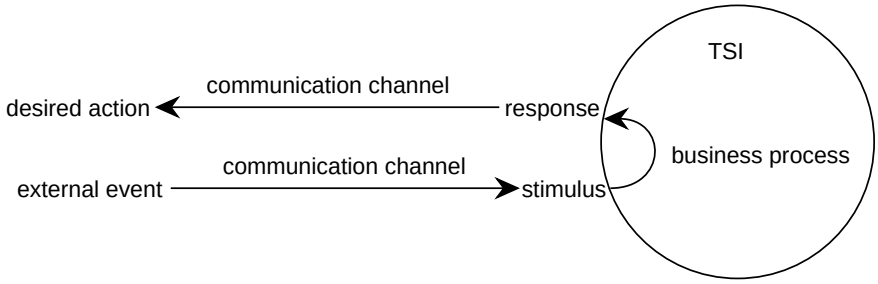

Figure 4.2: Stimulus-response processing by a business.

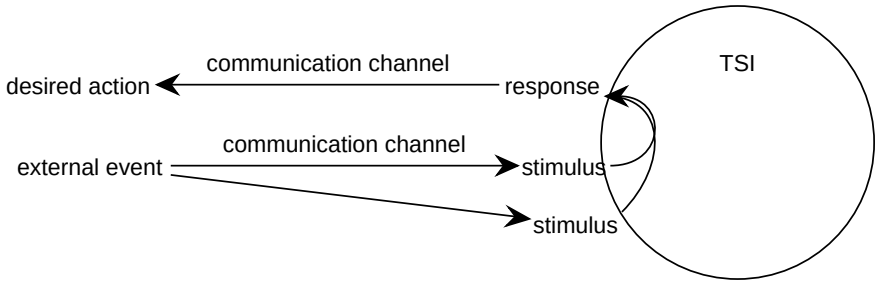

Figure 4.3: Multi-channeling.

environment. A business may learn of the occurrence of an external event through various channels (figure 4.3) and it may cause a desired action in the environment through various output channels. This is called multichanneling. A business process then contains parts dedicated to particular channels and other parts shared by all channels. TSI has one channel that connects it with its customers, travelers and third parties: Internet.

A special type of event that the business must respond to is the *temporal event.* A temporal event is a significant moment in time, e.g. the end of a day, the end of a quarter, the end of the period in which a customer should have made a payment, etc. There is no external entity that causes this event. The business discovers that an external event has occurred by looking at the clock regularly.

We define a *business process* as a sequence of activities that has a definite start (the triggering event) and that, when finished, has delivered a value (the delivered service. A process may consist of several activities. Each *activity* not only has a start but also a definite termination point. When terminated, it has contributed to part of the service to be provided by the process.

We now link stimuli to business processes and business functions. The meaning of an entry in the table of figure 4.4 is

> Stimulus s triggers business process b, which delivers service f to the environment.

The triggering stimulus starts the process. Possibly, during the processes, several stimuli and responses will be required and produced. The process ends when the value is delivered.

Figure 4.5 refines the services of TSI and figure 4.6 contains the process tables that links events to services via processes. Each entry of figure 4.6 gives the activities that may be performed as a result of receiving the event trigger. The result of performing one or more of these activities is the delivery of a service to a customer, namely the service listed in the leftmost column.

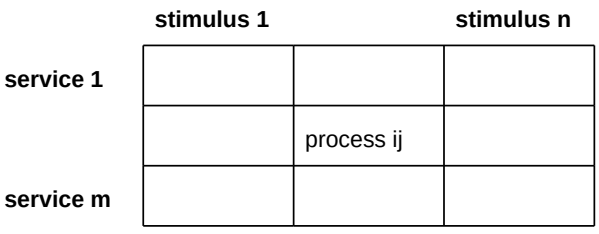

Figure 4.4: Format of a process table.

- Customer services:
  - Initialize web site and database: Create web site with brand of customer and create database with offerings.
  - Update web site and database: Maintain consistency with customer's situation.
  - Provide access and sales statistics: Periodically.
- Traveler services:
  - Provide information about holiday homes: Offer web-based browsing.
  - Offer reservation capability: Offer facilities to compose booking and place the booking and to cancel the booking.
  - Offer payment capability: Offer facilities to do initial and further payments. Monitor payments, produce reminders and reimburse if booking is canceled.
  - Offer feedback capability. Offer facility for traveler to lodge complaints and recount experiences.

Figure 4.5: Services provided by TSI. This is a refinement of the service list of figure 3.2.

Stimulus / **Service**	Traveller requests booking	Traveller pays	Final payment too late	Traveller cancels booking	Traveller complains	Traveller provides experiences	New customer acquired	Customer changes products	Time to provide supplier with occupation data
Booking	Compose booking. Check availability. Reserve booking.		Release booking.	Release booking.					
Payment	Accept payment	Accept payment. Clear payment	Send reminder. Reimburse.	Reimburse					
Feedback					Treat complaint	Collect feedback.			
Initialize customer web site							Create database and web site		
Maintain web site								Update	
Provide statistics									Derive data

Figure 4.6: TSI processes. The upper row lists events that lead to a stimulus. The leftmost column lists TSI services (the leaves of the function refinement tree). The entries lists the activities in the business process triggered by events and delivering services.

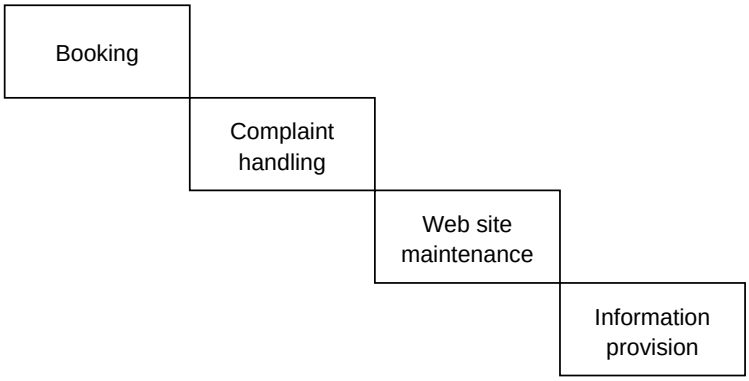

Figure 4.7: Subsystems of TSI. Each box corresponds to an activity that handles stimuli and provides services of TSI.

Chapter 5

Application Architecture

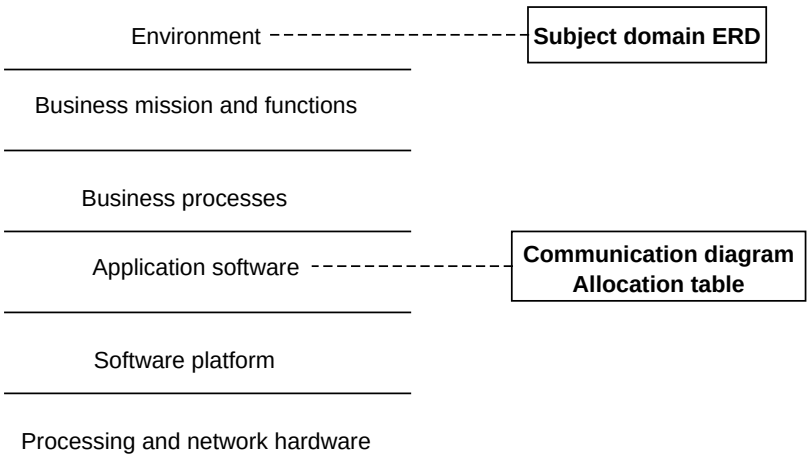

Figure 5.1: Focussing on the application architecture.

An easy way to start is to list the identified parts of TSI (figure 4.7) and postulate that we will need software systems to support these activities. Combined with context information, this gives us the architecture in figure 5.2. Adding databases and interface information, we get figure 5.3. The modules (boxes) all have access to all databases (parallel bars). The database contain data about the subject domain. Figure 5.4 represents the logical structure of the subject domain. This is an environment model, not a model of the databases; but it can be used as conceptual model of the logical structure of the data.

We can find a software decomposition by means of the following guidelines.

- Functional decomposition. One software component for each external service offered by the software.
- Device-oriented decomposition: one software component per external device interfaced with.
- Event-oriented decomposition: one software component per interesting subject domain event.
- Entity-oriented decomposition: one software component per interesting subject domain entity.

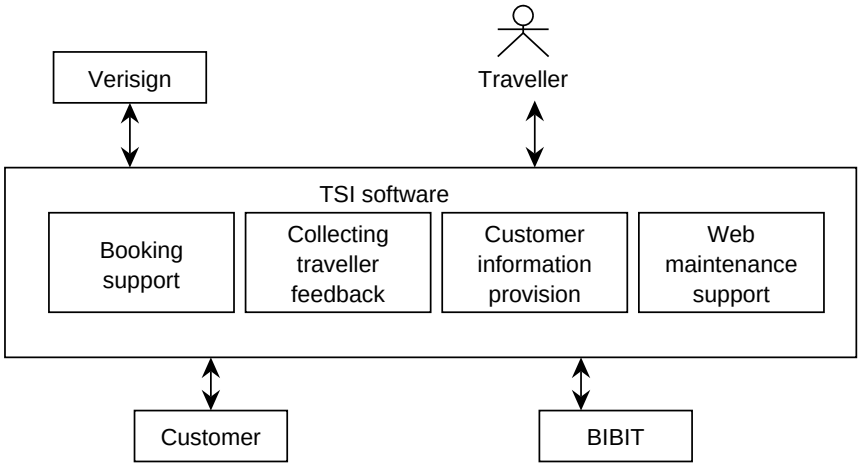

Figure 5.2: Application architecture version 0: Context and major subsystems.

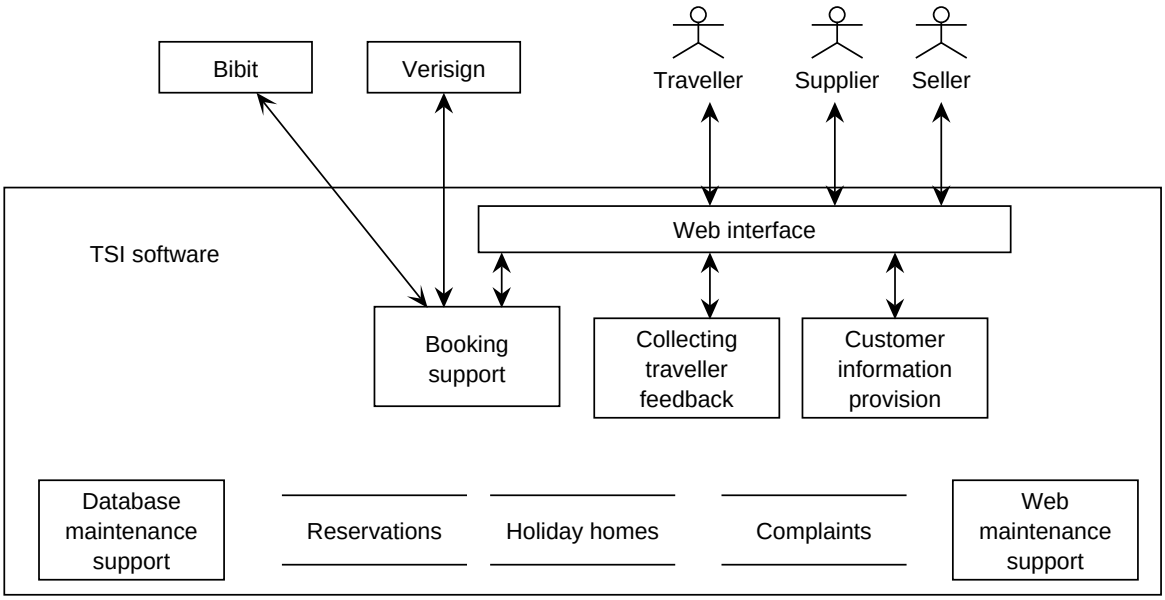

Figure 5.3: Application architecture version 1: Adding databases and interfaces to components.

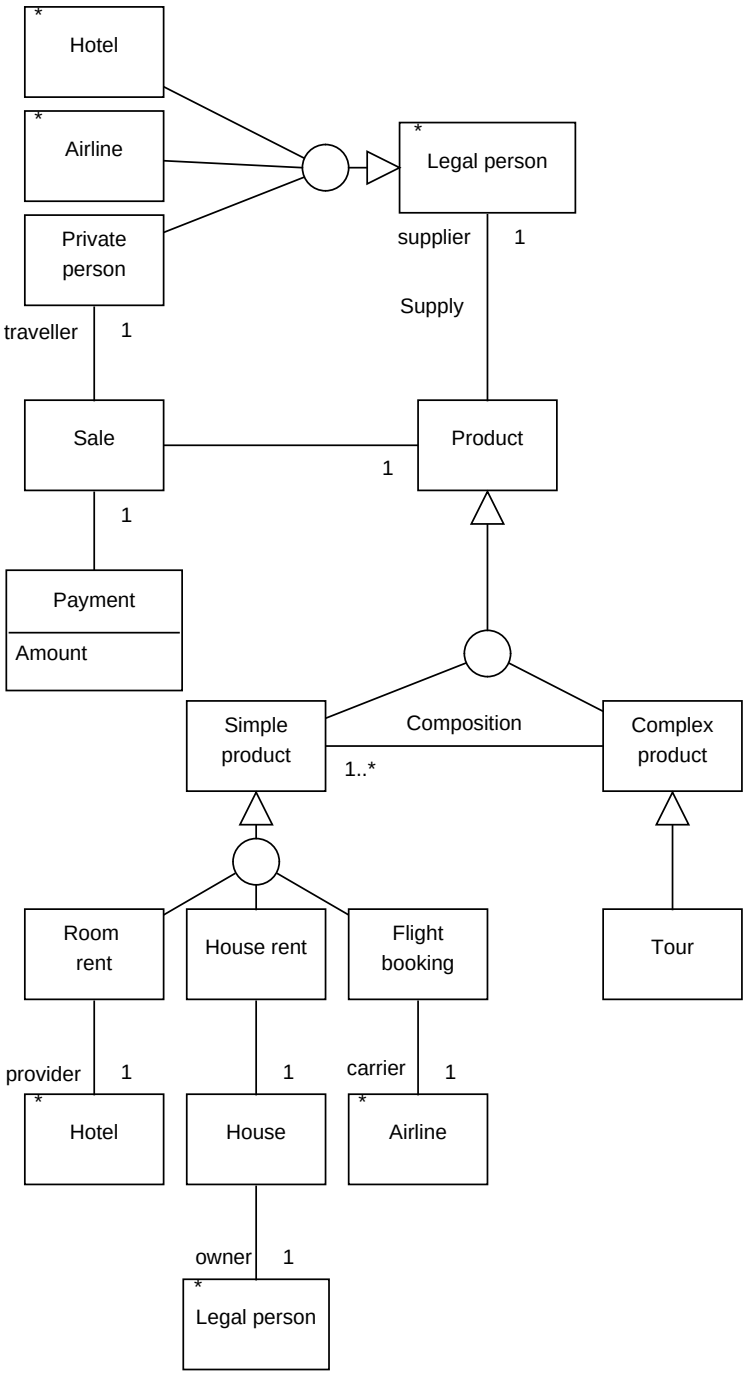

Figure 5.4: The subject domain of the TSI databases.

- Temporal partitioning: one component for stimuli that always occur in the same order.
- Real-time partitioning: one component per group of stimuli that occur at the same frequency.
- Real-space partitioning: one component per relevant geographical location (where a service must be offered).

Architectures can be evaluated by the following well-known criteria. These are not orthogonal to each other but they are worth mentioning separately.

- Modularity: close cohesion within components, loose coupling between components. There are several quantifications of the concepts of coupling and cohesion, such as control and data coupling.
- Changeability: How easily can the architecture be changed?
- Stability: How stable is the architecture under changes in the environment and in functionality?

We discuss options the TSI architecture in terms of these criteria.

Functional decomposition. According to this criterion, the components would correspond to the services in figure 3.3:

- Information provision component
- Booking component
- Payment component
- Traveler feedback component
- Database and web site initialization
- Database and web site update
- Maintenance of statistics

These components are not all modular, because there is a lot of interaction between booking and payment. Initialization and update of web site and database are also closely coupled.

Device-oriented decomposition. This involves defining a component for each connection technology with which the system must interface. Since we have no information about connection technology at TSI, we can give no example of this.

This is a good strategy to improve changeability of communication technology and to improve stability of the rest of the architecture. This guideline can be combined with other guidelines.

Event-oriented decomposition. This involves defining one component for each event to which the system must respond. The component would handle the complete response to this event. The following components would correspond to events:

- Booking request handling
- Payment handling: timely payments
- Payment handling: late payments

- Cancellation.
- Complaint handling
- Handle traveler experience report
- Register new customer
- Change customer products.
- Provide occupation data to customer.

This is not a modular architecture because many modules would be doing similar things.

Entity-oriented decomposition. Using this guideline, we define one component for each entity in the subject domain. So for each entity in figure 5.4, we would define a system component. This is what we do in database design, where each entity is represented by a database table, at least at the conceptual level.

Temporal partitioning. According to this criterion, we group stimuli that always (or usually) occur together and define one component for that. This leads to the following components:

- Booking and payments
- Traveler feedback
- Database and web page management
- Statistics

This is the partitioning that we chose.

Real-time partitioning. According to this guideline, we group stimuli that always occur at the same time, and deal with them with one component. There are no real-time characteristics that we can use.

Real-space partitioning. According to this criterion, we define one component per relevant geographic location. For example, we allocate software to nodes in a network close to the actors that must use this software. The Internet itself uses this criterion (web browsers in the traveler's PCs). This is really an implementation guideline, that tells us something about how to allocate software to physical computing resources.

The choices made. In figures 4.6 and 4.7, we combined functional decomposition (partitioning along the vertical axis of figure 4.6) with temporal partitioning (along the horizontal axis) to combine the advantages of functional decomposition, event partitioning and temporal partitioning.

Chapter 6

Network and Implementation Platform

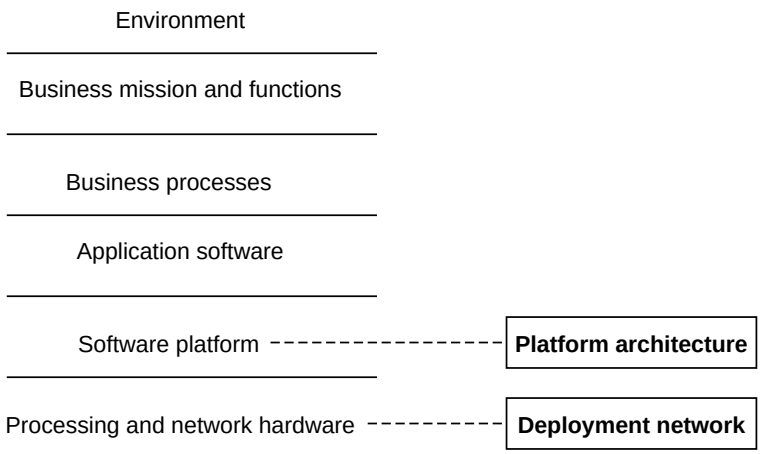

Figure 6.1: Allocating applications to a platform and network.

The deployment network (figure 6.2 gives the topology of physical resources. "Physical" means physical: Having a weight, size etc. The boxes in the diagram represent devices made of plastic, metals, silicium etc. The lines represent wires that you can touch.

Allocation of applications to nodes in the network can either be written in the diagram or else be represented separately in an allocation table of applications versus nodes. The table can represent a many-many relationship.

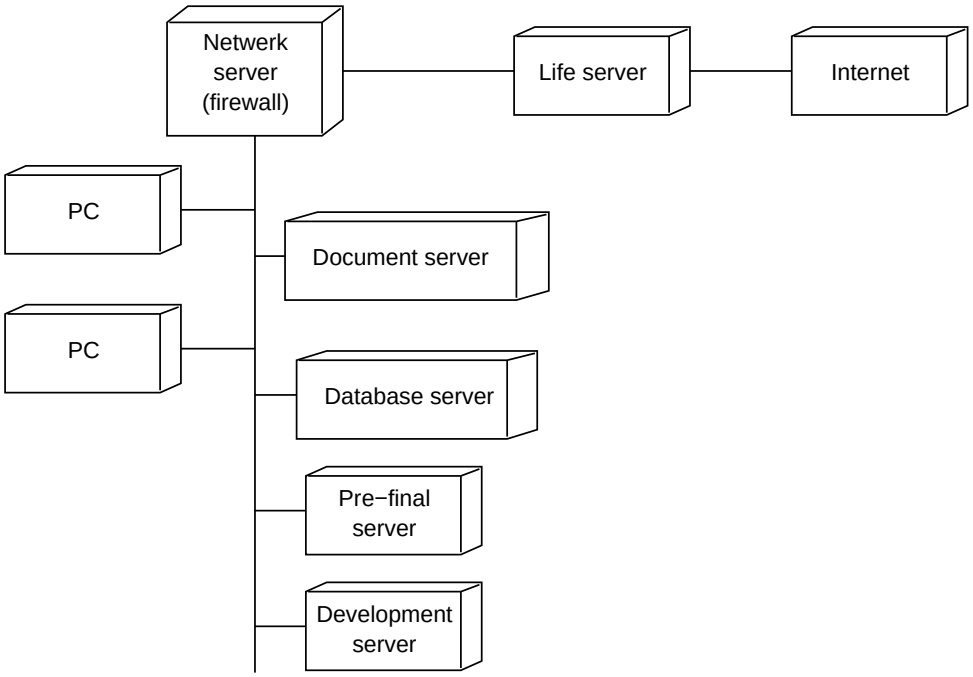

Figure 6.2: Network and implementation platform.

Chapter 7

Discussion

This report presents a case study to prove the feasibility of our architecture design ideas. The list of topics for further research is, as usual, longer than the list of achievements. TSI has asked us to upgrade the current architecture to deal with the following functionalities in a flexible way:

- Generalized search & book functionality for products other than homes, such as complete holiday packages, hotel rooms, boats, etc.
- Flexible configuration of connection to the back office systems of a supplier or seller.
- Flexible booking processes.
- Choose & buy products such as hotel vouchers and admission tickets.

More generally, here are some major topics to be researched:

- Make a list of consistency relationships between elements of the architecture specification.
- Can we provide tool support to manage traceability between the models.
- Can we develop guidelines for mapping a conceptual architecture to an implementation platform? Or for selecting an implementation platform for a conceptual architecture?
- Are there guidelines that relate required business performance to particular classes of architecture?
- Are there patterns of requirements and architectures that we can identify, and that have been proven to work well in different circumstances?
- Define an architecture to manage Service Level Agreements of a service provider: monitor service performance and relate this to contract with the service consumer.

We intend to look at some of these problems in future projects.

Bibliography

[1] R.J. Wieringa. *Design Methods for Software Systems: Yourdon, Statemate and the UML*. Morgan Kaufmann, 2002. Forthcoming.

Appendix A

Summary of Models

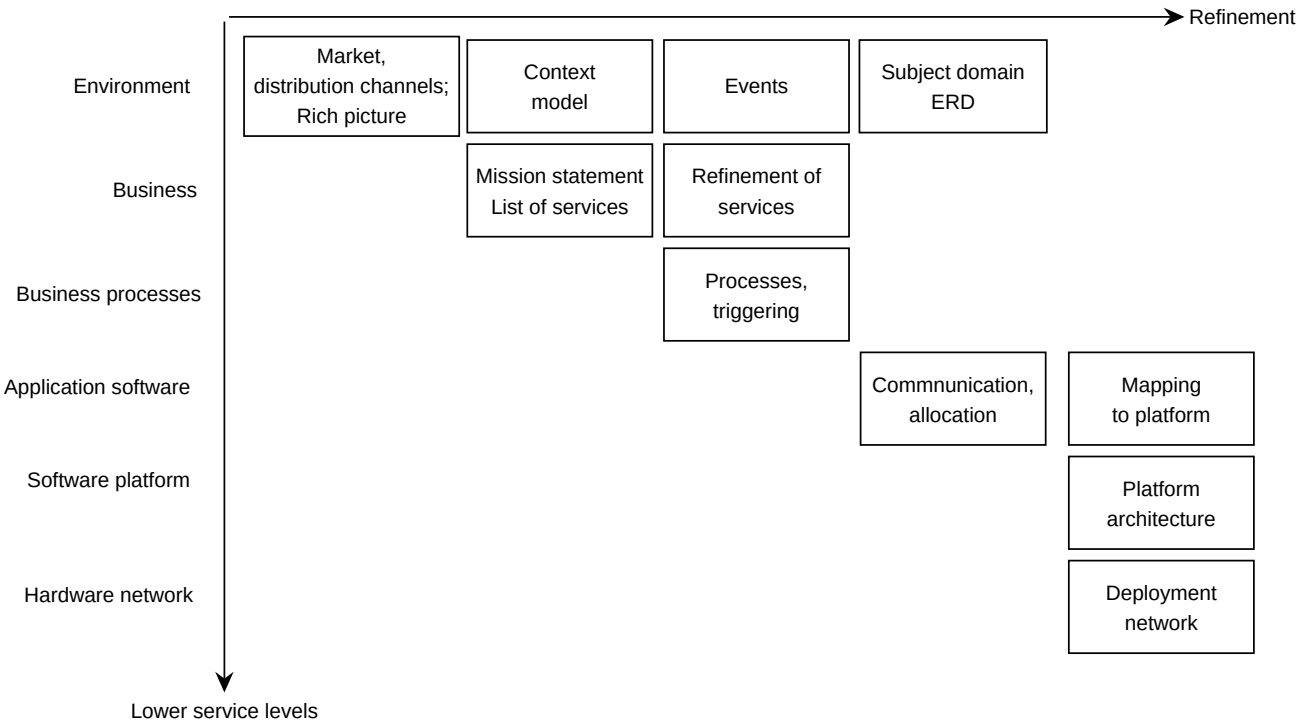

Figure A.1: Overview of models.