Chapter 3. Task And Domain: Methodology

3.1. Introduction

The logistics processes at Cabofa have been analyzed thoroughly. We will now focus our attention on the tasks and domain that are relevant for the production planners. Our purpose in examining the tasks and domain is to gain a better understanding of the goals of planning tasks, and of the knowledge within the domain that has been used to perform these tasks. This knowledge is necessary in order to be able to develop a prototype that increases the planning support at the fine planning stage.

Regarding the exploration of the tasks and domain, we will discuss (a) methods for acquiring knowledge, (b) theories from the literature that may help to interpret the data from the knowledge elicitation, and (c) application software that may assist in interpreting the data.

We used three methods in order to gain knowledge of the tasks and domain. First, we enhanced our knowledge skills and experience simply by analyzing the processes over a long period of time at the functional departments at Cabofa. Second, we conducted several structured and unstructured interviews with staff at the factory. We interviewed a wide range of staff, including production managers, employees on the work floor, the logistics manager and the driver in the warehouse. Third, we conducted thinking-aloud sessions with the production planners. During a thinking-aloud session, the production planner immediately verbalizes all his thoughts relating to a particular task. These sessions provided detailed information about a particular planning task in the form of a verbal report. Protocol analysis provides a general methodology for interpreting verbal reports. Section 3.2.1 discusses the main elements of protocol analysis and how to interpret data in a verbal report.

In addition to methods for acquiring knowledge, we also consider theories that may assist in interpreting data. The scientific literature suggests several theories for identifying the tasks and domain. We shall restrict our study to three pillars selected from the full spectrum: (i) knowledge acquisition methods, (ii) grounded theory, (iii) object-oriented analysis and design.

Knowledge acquisition methods have been developed in the field of knowledge engineering. The main goal of knowledge acquisition is to understand how
knowledge in the research subject is used. It is closely related to knowledge systems whereby the knowledge of experts in a particular domain is stored in a computer system. Grounded theory is a qualitative approach to developing theories from data. Object-oriented analysis and design (OOAD) propose straightforward methods for constructing a hierarchical (static) framework that assists in constructing a prototype task domain for the purpose of increasing planning support by means of a computer program.

Apart from theories and methods for acquiring and interpreting information, there are several tools that can be used during the analysis stage. We used a computer program, Atlas/TI, to interpret the qualitative data produced during the thinking-aloud sessions. The methodological foundations of Atlas/TI are derived from grounded theory, although this tool is not restricted to grounded theory alone. It is simply a tool for analyzing qualitative data in a consistent manner in order to construct a new theory or framework.

We will briefly discuss theories and methods for acquiring knowledge and interpreting data, as well as tools that may be of use during the analysis stage. The following section continues to present, in brief, the main ideas behind protocol analysis and the above-mentioned methodologies for acquiring knowledge and interpreting data. We will also explain the reasons for selecting certain theories or methodologies for the purpose of this research.

3.2. Methodological approaches

Methodological approaches provide guidelines for gaining additional insight into a particular subject. In this section, we discuss protocol analysis and three methods of interpreting data: knowledge acquisition (KA), grounded theory (GT), and object-oriented analysis and design (OOAD). The first methodological approach we will discuss is protocol analysis.

3.2.1. Protocol analysis

Protocol analysis consists of two stages. During the first stage, the production planner explains his task verbally. The speech and written texts that are produced are called verbal reports, or protocols. The second stage consists of analyzing the verbal reports. Elshout (1976) and Ericsson & Simon (1990) propose steps for analyzing the protocols so that a theoretical model can be constructed or tested.

The protocol analysis method originates from the beginning of the twentieth century, when Polya, Duncker, and Selz began to carry out introspective research. Prominent advocates of these types of method were Newell & Simon (1972). A seminal reference work in this field is Ericsson & Simon (1990). It provides a good theoretical background to protocol analysis and verbal reports as data.

There is a broad range of protocol-analysis applications, see for example Newell & Simon (1972), Bhaskar & Simon (1979), De Groot (1965), Peau & Murphy (1988), Hayes-Roth & Hayes-Roth (1979), Byrne (1977), and Kleiner & Drury (1998).

Page 50
In the past there was an intense debate on the validity of verbal reports. The debate centered on the question whether verbal reports could be considered as data (Leplat & Hoc, 1981; Nisbett & Wilson, 1977; Ericsson & Simon, 1990 and Praetorius & Duncan, 1988). Today, most researchers consider verbal reports as data, and therefore do not question their validity.

One of the main problems relating to protocol analysis is verbalization, which is a problem for some people. Some tasks may have such a high cognitive memory load that verbalization is not possible; the process is halted in favor of the problemsolving task.

Another restriction relates to the information available in the protocol. It can be expected that there is less information in the protocol in comparison to information processed in memory. This is also reported by people participating in a protocol analysis by Breuker et al. (1986).

To summarize, protocol analysis is a valuable method for uncovering the details of planning task. Problems relate to the completeness, correctness, and reliability of the verbal report, combined with the question whether the verbalization of the subjects’ thoughts exerts a significant influence on the execution of the task.

3.2.2. Knowledge-acquisition methods (KA)

Knowledge engineering involves acquiring knowledge from experts and storing it in knowledge systems. Several frameworks have been developed within this field. The most notable are Generic Tasks (Chandrasekaran & Johnson 1993), Componential Framework (Steels 1993), Protégé-I and II (Puerta et al. 1992) and the KADS and commonKADS-libraries (e.g. Schreiber et al., 1993; Valente & Breuker, 1998; Schreiber et al., 2000; Wielinga et al., 1993 and Studer et al., 1998).

The basic ideas underlying Generic Tasks (GT) can be characterized as follows: (a) GT is associated with the generic description of its input and output; (b) GT comes with a fixed scheme of knowledge types specifying the structure of the domain knowledge required to solve a task, and (c) GT includes a fixed problem-solving strategy that specifies the inference steps of which the strategy is composed, and the sequence in which these steps must be carried out.

Therefore, GT incorporates a fixed problem-solving strategy and a fixed collection of knowledge structures. Two main disadvantages of GT have been identified in the literature: (i) the notion of the task is conflated with the notion of the problem-solving method (PSM) used to solve the task, because each GT includes a predetermined problem-solving strategy; and (ii) the complexity of the proposed GTs varies widely, and the appropriate level of granularity remains open (Studer et al. 1998).

The componential framework identifies objects, relationships, properties and the attributes of objects. Two types of models have been applied: (1) domain model: this model is fixed in a particular application; and (2) case model: this model is built using the application.

A task is something that must be accomplished (Steels 1993). Models that have an impact on a task are called target models, while models that consult a task are
called source models. A task may also produce input or output from and to an interface, and may have one or more subtasks.

There are methods that specify how a task is performed: (i) task decomposition methods, (ii) solution methods, (iii) acquisition methods, (iv) inference methods, (v) learning or problem-solving methods, and (vi) presentation methods.

Protégé was developed to solve the problem of automating the construction of knowledge acquisition tools by reusing PSMs and ontologies. Ontology is defined as a formal, explicit specification of a shared conceptualization (Studer et al. 1998).

The key concepts in Protégé-II are tasks, mechanisms, and methods. A task is considered an activity, or an abstraction of an activity in the real world that accepts some form of input and produces some form of output. A mechanism is a procedure that completes or solves a task. It specifies how a task is performed. Methods are considered as the union of a number of tasks.

Protégé-II consists of a library of mechanisms and a user-interface management system. It supports the PSMs and the task model that is formulated by the knowledge engineer. All these elements are necessary for constructing a knowledge acquisition tool.

The aim of Protégé is to build knowledge tools that are domain-independent. However, the implementation presents interfaces that are task-dependent, domain-dependent and user-dependent. This is a clear paradox (Puerta et al. 1992).

The commonKADS methodology has been used to construct a library of re-usable problem-solving components. The role of the library is to support the knowledge engineering process (Valente & Breuker 1998). CommonKADS consists of two elements: (i) a framework for implementing domain-independent specification of knowledge in combination with re-using knowledge models for frequently recurring tasks; (ii) a life-cycle model indicating the phases, activities and products that are relevant for a knowledge-based system (KBS) project.

The commonKADS methodology is based on a number of fundamental principles (Schreiber et al. 2000): (a) knowledge engineering is not a method for ‘tapping knowledge from the expert’s mind’, but involves constructing different aspect models of human knowledge; (b) knowledge modeling should concentrate first on the conceptual structure of knowledge, and at a later stage on programming details; (c) knowledge has a stable internal structure that can be analyzed by distinguishing specific knowledge types and roles; and (d) a knowledge project must be managed by learning from experience – i.e. by means of a ‘learning spiral’.

The development of a KBS is supported by the commonKADS model suite. The model suite contains six models:

- **Organization model.** The organization model supports the analysis of the major features of an organization.
- **Task model.** The task model analyzes the global task layout, its inputs and outputs, preconditions and performance criteria, as well as the resources and competences required.
- **Agent model.** The agent model describes the characteristics of agents (those who execute a task).
• Knowledge model. The knowledge model gives a detailed explanation of the types and structures of the knowledge used to perform a task.

• Communication model. The communication model analyzes the communicative transactions between the agents involved.

• Design model. The design model comprises the technical system specification in terms of architecture, implementation platform, software modules, etc.

The commonKADS methodology describes a range of activities that provide support for the knowledge engineer. The details are presented in Schreiber et al. (2000), which guides the knowledge engineer through each stage. This is beyond the scope of our study.

We have briefly outlined the main features of GT, Componential Framework, Protégé, and commonKADS. We selected the commonKADS framework in order to apply it to the knowledge elicitation process.

For the purpose of this study, the task model and knowledge model are the most relevant commonKADS models. The other types of model have not been applied in the context of commonKADS. For example, we did not use the organization model in Chapter 2, but instead applied general logistical-diagnosis methods.

The emphasis on tasks and the knowledge required to perform them is an important element in all the concepts mentioned above. In general, distinction is made between the following types of knowledge (Wielinga et al., 1993 and Schreiber et al., 2000):

• strategic knowledge. This provides knowledge for a very flexible form of problem-solving, for example by dynamically rescheduling the set of goals when impasses are detected due to information being unavailable, or due to conflicts in the reasoning process (Tansley & Hayball 1993).

• task knowledge. Task knowledge specifies tasks and relates to the goal of a task, as well as to activities that contribute to achieving the goal.

• inference knowledge. Inference knowledge specifies basic inferences that can be made from the domain knowledge. Inferences are the building blocks for the reasoning mechanism. The main elements of inference knowledge are inferences, knowledge roles, and transfer functions.

• domain knowledge. Domain knowledge expresses relevant types of knowledge and objects regarding the systems to which a task relates.

Although emphasis is placed on task, inference and domain knowledge, strategic knowledge is also considered important in the following chapters.

We would like to clarify the distinctions between the types of knowledge by using the following example from Schreiber et al. (2000). Figure 12 shows a schematic representation of a diagnostic task.
The example shows the diagnosis of a medical disease. The doctor will hypothesize a diagnosis based on the symptoms presented. The task is therefore the diagnosis, the symptoms found are in the domain, and the inference layer provides the connection between task and knowledge.

Similar templates have been developed for the planning and scheduling tasks (see also Figure 58 on page 168). Basic inferences within the scheduling task are Select, Decompose, Match and Specify; within the planning task Select, Assemble, Compare, Abstract, Decompose, and Specify (see Tansley & Hayball, 1993). A detailed representation of a planning or scheduling task in the same manner as Figure 12 is given in Figure 13.
The same type of hierarchy as presented in Figure 12 can be found in the tasks and domain at Cabofa. This is part of detailed discussion in Chapters 4 and 5.

Another important concept is task. In commonKADS, it is defined as a human activity designed to achieve a particular purpose. Figure 14 shows how commonKADS views the relationship between task in the environment and corresponding elements (Figure taken from Schreiber et al., 2000).

![Task model diagram]

**Figure 14: Task model**

A task is subsection of a business process that:
- represents a goal-oriented activity adding value to the organization;
- deals with input and delivers desired output in a structured and controlled way;
- consumes resources;
- requires (and provides) knowledge and other competences;
- is carried out in accordance with given quality and performance criteria;
- is performed by responsible and accountable agents.

CommonKADS has evolved over the past ten years. A complete library is now available containing different types of models and related templates. Although the conceptual model and ideas underlying commonKADS can be generally applied to the cardboard industry, we decide to use our own systematic approach.

It is motivated by the following arguments. First, we do not need all models from the model suite. The agent model and communication model are not relevant, because we do not consider distinct agents, but only a human decision maker that requires planning support. Second, commonKADS’s point of departure is the task. Although the task is very important, too much emphasis on it may neglect the systems
Third, logistics and organization science provide more mature methods to analyze the organization and production planning and control from a systems perspective.

Thus, we do not apply methods from commonKADS only; our approach combines elements from it with methods from Grounded Theory and Object-Oriented Analysis and Design. We continue with a brief description of Grounded Theory.

3.2.3. Grounded theory (GT)

The importance of empirical observation for formulating theories in the social sciences has been emphasized by Glaser & Strauss (1967). They introduced the concept of grounded theory, which is developed through induction based on observation of the phenomenon in question (Strauss & Corbin 1990). In their view, the discovery and development of new theories are the result of observation rather than abstract thought. They consider the development of theory as a process whereby new concepts are formulated on the basis of observations. As the theory develops, it continues to have a double function: interpretation and as an explanation of reality (Den Hertog 1995). Finally, the theory must contribute to the actual understanding of the phenomena.

According to Strauss & Corbin (1990), good grounded theory satisfies the following four criteria: (1) it should fit the phenomenon, (2) it should aid understanding and be comprehensive, (3) it should provide a general context, and (4) it should provide control, in the sense of stipulating the conditions under which the theory applies.

The formulation of a new theory begins by coding the data and formulating relationships between the coded objects. Coding the data involves three processes that may overlap: open coding (whereby data is broken open to identify relevant categories); axial coding (whereby categories are refined, developed and related), and selective coding (whereby the “core category” is identified and related to other categories).

We apply the following two results from the methods produced by grounded theory: groundedness and density. Groundedness relates to the code frequency, whereas density relates to the relation frequency. A detailed analysis of the planning and scheduling domain, with the application of these results, is one of the central concepts in Chapter 5.

There are many references in the literature to Grounded Theory as the main framework for research design. A detailed description of grounded theory is beyond the scope of this dissertation, so we will not discuss the fundamental building blocks of grounded theory. For more about grounded theory in relation to the use of computer programs that follow this approach, and about the status of grounded theory as a theory, we refer to Kelle (1997), and Tashakkori and Teddlie (1998).

The third approach employed in the research highlights object-oriented analysis and design framework, and is discussed in the following section.
3.2.4. Object-oriented analysis and design (OOAD)

Object-oriented analysis and design (OOAD) methods evolved in the 1980s and 1990s. Their goal is to provide a consistent and hierarchical explanation of the domain. There are two types of object-oriented (OO) method: object-oriented analysis (OOA) and object-oriented design (OOD). Analysis involves problem definition/modeling, while design focuses on solution specification/modeling. The differences between analysis and design methods will not be discussed here.

The following authors, among others, have developed several types of object-oriented analysis and design framework: Booch (1994), Martin (1987), Martin & Odell (1995), Shlaer & Mellor (1988), and Yourdon (1994). An overview of some of these frameworks is given in Monarchi & Puhr (1992) and Champeaux & Faure (1992).

Coad & Yourdon (1991) define an object as an abstraction of something in the domain or an implementation of a problem that reflects the ability of a system to store information about it and/or interact with it; an encapsulation of Attribute values and their exclusive Services.

Walker (1992) describes topics on which OOAD should focus: (i) identifying objects and defining classes; (ii) the hierarchical organization of classes; (iii) the re-use of classes; and (iv) the construction of application frameworks from class libraries. According to Booch (1994), any design method should address at least the following topics: (a) abstraction, (b) encapsulation, (c) modularity, (d) hierarchy, and (e) typing. Abstraction involves stratification with respect to the application domain or classes. Encapsulation is the combining of data and process within an object. Modularity involves the allocation of objects and functionality to compilation units; it is related to the decomposition of the domain into sub-domains. Hierarchy is important with regard to the object. So-called ‘children objects’ inherit the properties of their parent object. Typing is a way of categorizing values that are produced by computing activities.

Domain modeling, the central aspect of most KA and GT methods, generally uses an OO framework. The general principles of OO have therefore been covered in this discussion.

3.3. How to apply these theoretical frameworks in Chapter 4 and 5

The conceptual framework we use is closely related to the commonKADS framework. We distinguish between four types of knowledge: strategic knowledge, task knowledge, inference knowledge, and domain knowledge. Chapter 4 provides a systematic overview of the planning tasks within the planning and scheduling domain as a whole, and shows how planning tasks relate to task and inference knowledge. Chapter 5 places greater emphasis on the domain-knowledge layer.

We derived the objects and corresponding properties and events from ‘thinking-aloud sessions’ conducted with the production planners. The results of the thinking-aloud sessions are verbal reports that can be considered as qualitative data from
which a theory can be derived (in our case a consistent framework for domain knowledge). We applied the results from Grounded Theory (groundedness and density) to derive the prominent codes that correspond to objects and their properties and events that have been processed intensively within the planning and scheduling domain. Another projection was also used which we call the domain view. We analyzed the data on the basis of the restricted set of objects that generally exist within a planning or scheduling domain.

In that sense, there is a strong connection between Chapters 3, 4, and 5: each chapter deals with the tasks and domain that need to be identified in order to develop a planning tool that provides increased planning support in the domain.

The following chapter focuses on planning tasks by systematically analyzing what they involve. Please note that the discussions relates to the task and inference knowledge layer of commonKADS.