

Overview of the Step-by-step Procedure

This chapter introduces a procedure of 47 steps that covers the activities required for a successful knowledge acquisition project. I start by introducing three essential aspects of the procedure: knowledge capture (techniques to use when seeing experts), knowledge analysis (identifying the elements required to build the k-base) and knowledge modelling (creating different ways of editing and viewing the k-base). I will then walk you through a complete project that uses the 47-step procedure. The chapter ends with a list showing how the steps in the procedure contribute to the usefulness, usability and use of the end-product.

2.1 Knowledge Capture

2.1.1 Knowledge Capture Techniques

Key parts of a knowledge acquisition project are the encounters with experts. During the stages that capture domain knowledge, these sessions are concerned with two things: (i) **Eliciting** knowledge, *i.e.* capturing knowledge that is not in the k-base; (ii) **Validating** knowledge, *i.e.* checking that knowledge in the k-base is correct, complete and relevant to the project

What techniques can be used for eliciting and validating knowledge? As introduced in Chapter 1, we need to do more than ask an expert a few questions and expect knowledge to pour forth. We need to deal with memory effects and communication difficulties. We need to deal with our own lack of understanding. We need to deal with experts having different opinions.

To do this requires having a range of techniques that are based on a clear understanding of the psychology of expertise. Such a body of techniques has been developed over the past 25 years by knowledge engineers, and includes techniques that have undergone critical analysis using controlled experiments (Shadbolt, 2005). Emerging from this work are three groups of techniques: interview techniques, modelling techniques and specialised techniques.

2.1.1.1 Interview Techniques

Interview techniques involve questioning the experts. They are good for eliciting basic knowledge but not good for validating knowledge. There are 3 variants: unstructured interview, semi-structured interview and structured interview.

- The unstructured interview has very little planning and is a freeform chat with the expert. This can be used in the early stages of elicitation to get some basic knowledge of the domain but is not normally used for most elicitation sessions, as it is not very efficient.
- The semi-structured interview is the main technique for eliciting explicit knowledge. It uses a pre-defined set of questions that are sent to the expert beforehand, and supplementary questions that are asked at the interview. Steps 15 and 16 of the 47-step procedure are specifically concerned with preparing and conducting semi-structured interviews.
- The structured interview uses a pre-defined set of questions and no supplementary questions. It often involves a questionnaire that is filled-in at the session. This is usually preferable to sending questionnaires to people, as they rarely respond to them. The structured interview has its uses but it is not a main technique used in the 47-step procedure.

2.1.1.2 Modelling Techniques

Modelling techniques involve the use of knowledge models with experts. Knowledge models, or **k-models** for short, are ways of viewing the k-base using different forms of diagrams and matrices (see Section 2.3 for a full description and many examples). K-models can be used in three ways:

- A rudimentary k-model is shown to an expert and he/she is prompted to modify the errors and fill-in the gaps. This is used in Step 29 of the procedure to validate the knowledge from interviews and to prompt the expert to add more detailed knowledge.
- A complete k-model is shown to the expert who provided the knowledge or a secondary expert. The latter is the main way to perform cross-validation in Step 36 of the procedure.
- A k-model is created from scratch on a blank sheet of paper or blank computer screen. This technique can be used to capture basic knowledge (e.g. use of a timeline in Step 16) or to elicit deeper knowledge (e.g. the use of a concept map in Steps 30 and 35).

2.1.1.3 Specialised Techniques

Specialised techniques (a.k.a. contrived techniques) involve methods developed by psychologists for probing the expert's mind for deep, tacit knowledge. They tend to involve the expert performing a task that will expose and reveal knowledge that is difficult to articulate. These tasks come in two varieties: (i) Tasks that the expert normally performs, such as in the commentary technique; (ii) Tasks that have been devised to probe the expert, such as in the concept sorting and triadic elicitation techniques. A description of the main specialised techniques can be found in Step 35 (on pages 125-132).

2.1.2 Comparing the Capture Techniques

Having briefly surveyed the main classes of capture techniques let me now turn to some of the similarities and differences between them.

2.1.2.1 Similarities Between the Capture Techniques

All of the capture techniques do three main things:

1. They **focus** the expert on the knowledge that is required, and discourage the expert from wandering away from that;
2. They help the expert to **recall** knowledge by providing prompts or by approaching the knowledge from different angles;
3. They help the expert to **explain** what he/she knows in a clear way

2.1.2.2 Differences Between the Capture Techniques

One of the main differences between the techniques is that interview techniques and specialised techniques are used to elicit knowledge but not to validate knowledge whereas modelling techniques can do both. Another difference is the type of knowledge that each technique is best at eliciting. Using the two main dimensions of knowledge described in Chapter 1, Figure 2.1 shows which techniques are most appropriate to use for particular types of knowledge.

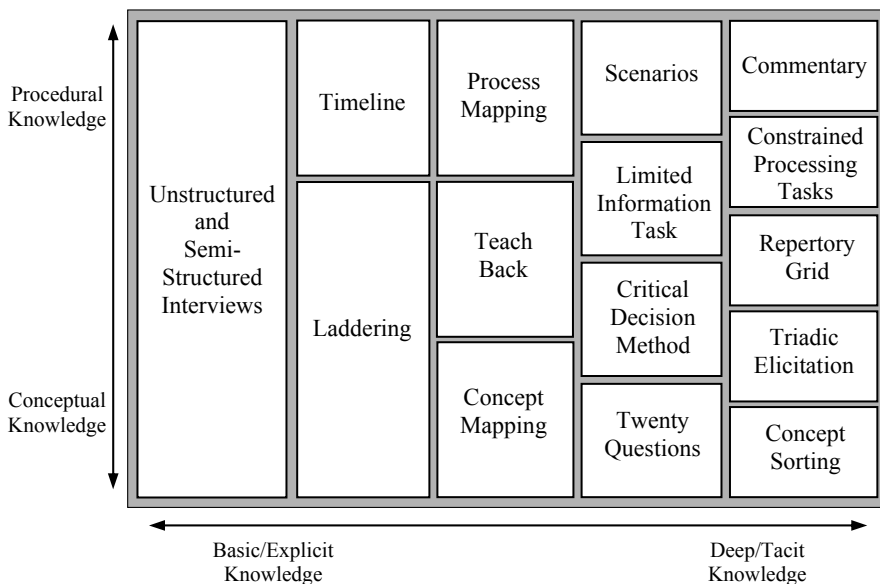


Figure 2.1. Techniques for capturing different types of knowledge

In Figure 2.1, interview techniques are shown on the left indicating that they are most effective for basic, explicit knowledge that is both conceptual and procedural. Modelling techniques (timeline, laddering, process mapping, teach back and concept mapping) are shown in the centre of the horizontal axis as they are used to capture more detailed knowledge than interviews. Timeline and process mapping are at the top indicating that they are most useful for procedural knowledge, and concept mapping and laddering are towards the bottom showing that these are more effective for conceptual knowledge. Specialised techniques are shown on the right as they are used to capture deep, tacit knowledge. An explanation of each of the specialised techniques is contained in Step 35 (see Section 5.6).

2.2 Knowledge Analysis

Knowledge analysis is an activity performed by the knowledge engineer after he/she has had a knowledge elicitation session with an expert. It is concerned with identifying elements of knowledge that will be entered into the k-base to form its structure and main components. These elements are those that will be used as building blocks to form all of the k-models. There are four important elements (a.k.a. knowledge objects) that can be identified during knowledge analysis: concepts, attributes, values and relations. Let us take a look at each of these.

2.2.1 Concepts

Concepts are the things that constitute a domain. Some of the main types are:

- Physical concepts, *e.g.* products, components, machines;
- Pieces of information, *e.g.* ideas, plans, goals, requirements;
- Sources of information, *e.g.* documents, databases, websites;
- People and roles, *e.g.* domain experts, roles of domain experts;
- Organisations and groups, *e.g.* producers, suppliers, divisions;
- Areas of knowledge, *e.g.* Physics, Corporate Law, Systems Analysis;
- Functions, *e.g.* the purpose of components and roles;
- Tasks, *e.g.* the activities performed by experts;
- Issues, *e.g.* problems, solutions, advantages, disadvantages;
- Physical phenomena, *e.g.* physical mechanisms and forces;
- Others, *e.g.* materials, behaviours, constraints, states

Concepts form the main structure of the k-base. The other elements of the k-base are there to describe the concepts. This is done in two ways: (i) The properties of concepts are described using **attributes** and **values**; (ii) The ways in which pairs of concepts relate to one another are described using **relations**.

2.2.2 Attributes

Attributes are the qualities or features belonging to a class of concepts. In other words, they are the ways in which we see concepts as being different from each other. Some examples of attributes are:

- Attributes of physical objects, *e.g.* weight, shape, age;
- Attributes of information, *e.g.* source, format, importance;
- Attributes of people, *e.g.* gender, age, salary, personality;
- Attributes of organisations, *e.g.* size, turnover, product range

2.2.3 Values

Values are the specific qualities or features belonging to a concept that distinguish it from other concepts. Each value is always associated with an attribute. Some examples of values and their associated attributes are:

- The values *red*, *green* and *black* are associated with the attribute *colour*;
- The values *light* and *very heavy* are associated with the attribute *weight*;
- The values *0*, *1* and *4* are associated with the attribute *number of children*;
- The values *3.7* and *8.2* are associated with the attribute *acid-test ratio*;
- The values *4-legged marsupial* and *Technique for calculating the roots of a quadratic equation* are associated with the attribute *glossary entry*

As can be seen in these examples, values come in different varieties. Some values are adjectives, some are numbers and some are sentences. Others can be paragraphs of text or chunks of hypertext code that include hyperlinks, images and pictures. To reflect these varieties, we often group attributes into classes, such as:

- Categorical attributes for values that are adjectives, *e.g.* *red* and *heavy*;
- Numerical attributes for values that are numbers, *e.g.* *1* and *8.2*;
- Text attributes for values that are one or two sentences;
- Hypertext attributes for values that are chunks of hypertext;

2.2.4 Relations

Relations are the ways in which pairs of concepts are associated with one another. For example, what is the relation between ‘cheese’ and ‘food’? Obviously it is that cheese **is a type of** food. This type of relation tells us what class something belongs to, and is usually shortened to ‘**is a**’. This is the most important relation. Other important relations are:

- ‘has part’, *e.g.* car – has part – engine;
- ‘performs’, *e.g.* knowledge engineer – performs – create concept tree;
- ‘followed by’, *e.g.* create model – followed by – validate model;
- ‘requires’, *e.g.* electric torch – requires – batteries;
- ‘causes’, *e.g.* contact with acid – causes – corrosion;
- ‘produces’, *e.g.* KA project – produces – knowledge web

Every relation can also have an ‘inverse’ that goes in the opposite direction. For example, the inverse of ‘has part’ is ‘part of’, and the inverse of ‘performs’ is ‘performed by’. Some k-bases will include the inverses of some relations and others will not. It depends on how you want to store and model the knowledge.

To clarify some terminology: when a relation is connecting two concepts, then it is referred to as a ‘relationship’ or a ‘triple’ (because there are 3 elements in it).

2.3 Knowledge Modelling

Knowledge modelling involves the creation and use of knowledge models (or **k-models** for short). K-models are ways of viewing the knowledge contained in a k-base. The k-base is a confusing mass of interconnected knowledge, so it is important that we have ways of viewing parts of it using simple and clear views. Each k-model should be thought of as providing a different perspective or different viewpoint from which to see different aspects of the k-base.

As mentioned previously, k-models are vital for helping to elicit knowledge from experts and for validating the knowledge that has been captured. K-models are also used in the end-product to convey knowledge to end-users.

Some of the main k-models are trees, matrices, maps, timelines, frames and k-pages. Information and examples of these are given in the following sections.

2.3.1 Trees

A tree is a diagram that shows a hierarchical arrangement of nodes. Each node represents a concept in the k-base and each link represents a relationship between a pair of concepts. Some important types are concept tree, composition tree, process tree, attribute tree, cause tree and mixed tree. These are described below.

A **concept tree** is the most important tree, and possibly the most important k-model. It is characterised by having every link as the ‘is a’ relation. Hence, it shows what type of thing everything in the k-base is. This form of knowledge is called a taxonomy and is a key aspect of the k-base. An example of a concept tree is shown in Figure 2.2 (where we see that dolphins are a type of whale).

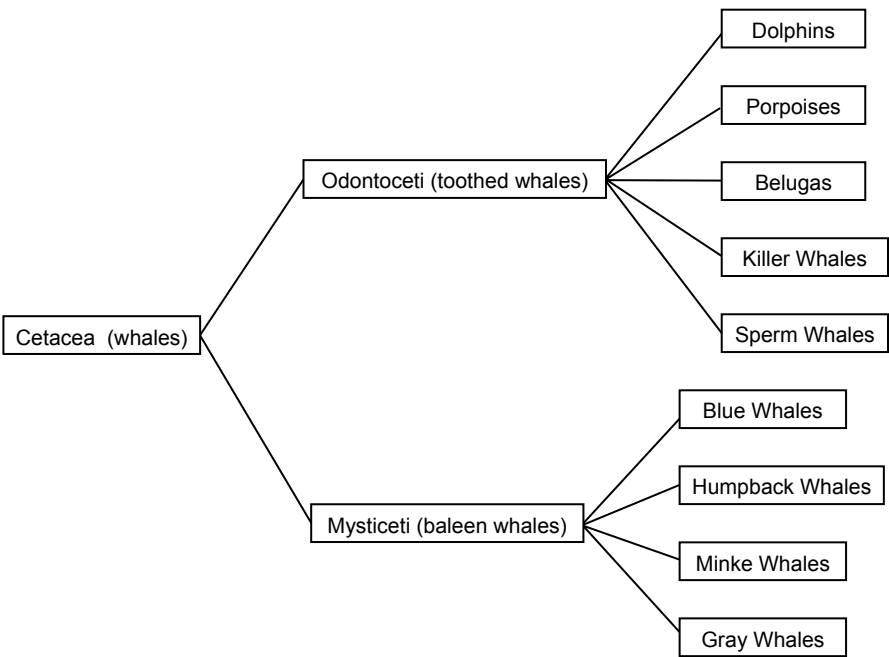


Figure 2.2. Example of a concept tree

A **composition tree** is characterised by having all the links as the ‘has part’ relation. This is used to show the components and sub-components of a concept such as a complex product, a document or an organisation. An example of a composition tree is shown in Figure 4.8 (on page 99).

A **process tree** is a special form of composition tree in which all the nodes are tasks. Hence, it shows how a complex task is composed of sub-tasks and sub-sub-tasks, *etc.* An example is shown in Figure 4.10 (on page 106).

An **attribute tree** shows the attributes and values that describe the properties of certain concepts in the k-base. An example is shown in Figure 4.9 (on page 102).

A **cause tree** is characterised by having the ‘caused by’ relationship for all links. It is used in projects that require knowledge to be captured of how experts diagnose a problem.

A **mixed tree** is a tree that contains more than one type of relation. An example is shown in Figure 2.3. The relations shown on the tree in Figure 2.3 are ‘is a’ for the black lines, ‘has exhibit’ for the black arrowed lines and ‘painted by’ for the dashed lines.

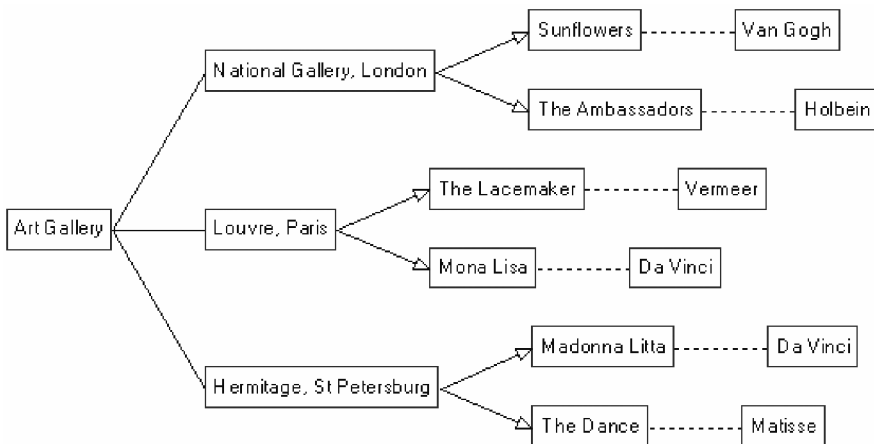


Figure 2.3. Example of a mixed tree

Note by convention ‘is a’ is read from right to left (*i.e.* is directed towards the highest, root node of a tree), whereas all other relations are read from left to right (*i.e.* directed away from the root node).

2.3.2 Matrices

There are two main types of matrix, an attribute matrix and a relationship matrix.

An **attribute matrix** is a way of showing the properties (attributes and values) of a set of concepts. It does this by presenting a matrix with concepts on the vertical axis and attributes/values along the horizontal axis. An example of an attribute matrix is shown in Figure 2.4, which shows the properties of a set of drinks.

		transparency			alcohol level		serving temp		fizziness	
		opaque	semi opaque	transparent	alcoholic	non-alcoholic	served hot	served cold	fizzy	not fizzy
drinks	lemonade			■		■		■	■	
	water			■		■		■		■
	vodka			■	■			■		■
	lager		■		■			■	■	
	coffee	■				■	■			■

Figure 2.4. Example of an attribute matrix

A **relationship matrix** shows two sets of concepts related to one another using a specified relation. The cells in the matrix show which pairs of concepts have the specified relationship. Figure 2.5 shows an example of a relationship matrix. This matrix shows people linked to areas of knowledge using the ‘has expertise’ relation, *e.g.* ‘Jane Lenton – has expertise – financial management’.

		people							
		Jane Lenton	Tom Mansfield	Ian Chesterfield	Dave Newark	Natasha Dacha	Maurice Leeds	Simon Bradford	Carrie Lincoln
areas of knowledge	financial management	□					□		
	resource management		□			□			□
	outsourcing	□	□	□				□	
	business transformation				□			□	
	business analysis	□	□	□			□		

Figure 2.5. Example of a relationship matrix

2.3.3 Maps

A map is a diagram that shows an arrangement of nodes linked by arrows. Each node represents a concept in the k-base and each link represents a relationship between a pair of concepts. Hence, a map is like a mixed tree that need not be hierarchical. The most important types are concept maps and process maps (see below). Other diagram formats can also be used, such state diagrams (see Figure 5.1 on page 132).

A **concept map** shows a variety of concepts connected by a mixture of different relations. An example is shown in Figure 2.6. Concept maps can come in different varieties, such as hierarchical concept maps (similar to mixed trees) and those that restrict the concepts and relations that are shown. Other names for a concept map are conceptual map, entity-relation diagram and semantic network.

The use of concept maps has been advocated as a comprehensive technique for capturing knowledge from experts (Zaff *et al.*, 1993; Cañas *et al.*, 2004).

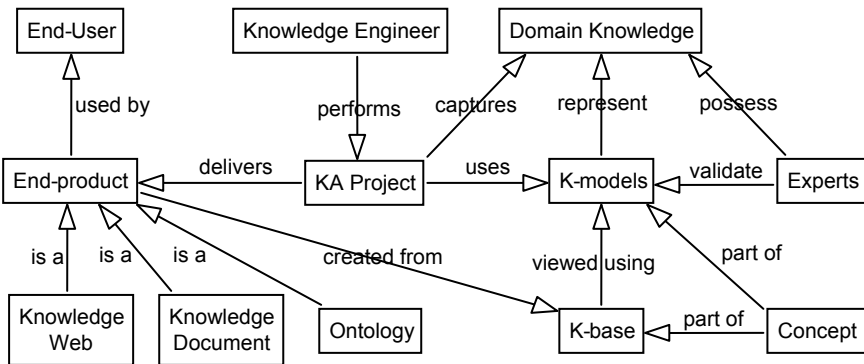


Figure 2.6. Example of a concept map

A **process map** shows the way a task (process, activity) is performed. The main elements on a process map are the sub-tasks of the task that is being modelled. These sub-tasks are placed on the map in the order in which they are performed. Links between tasks represent the ‘followed by’ relation. Other concepts can be shown on the process map with links to the task nodes using arrows. These can include: the resources required to perform each task; the products of each task; the triggers that cause a task to start; the people, roles or things that perform each task; the decision points that affect which tasks are performed. An example of a process map is shown on Figure 4.11 (on page 107).

2.3.4 Timeline

A timeline is a diagram that shows time along the horizontal axis and contains concepts as nodes. The width of each node shows when the concept starts and finishes. This can be used to show the phases of a project or the order of events or tasks. It is a simple representation that is often used in the early stages of

knowledge elicitation to capture the basics of processes from the expert (see Steps 15 and 16). An example of a timeline is shown in Figure 2.7.

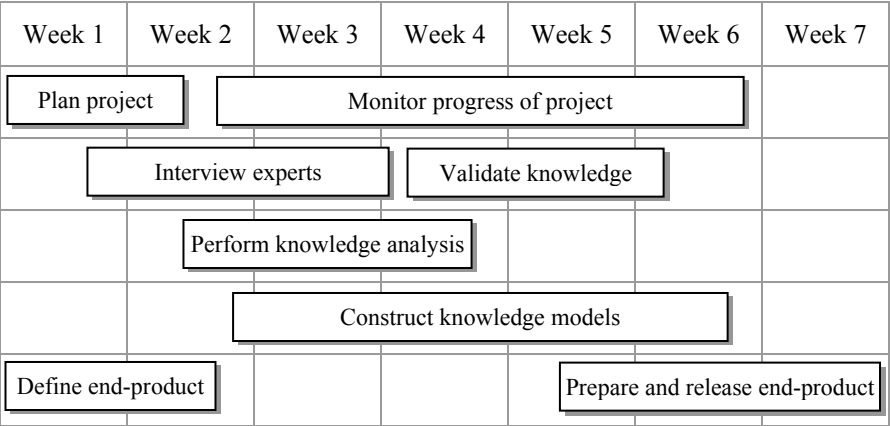


Figure 2.7. Example of a timeline

2.3.5 Frame

A frame is a simple 2-column table that shows the properties (attributes and values) of a concept. It is arranged so that the attributes associated with the concept are shown in the left-hand column and the corresponding values are shown in the right-hand column. Table 4.3 (on page 101) shows two frames, one showing the attributes and values of coffee the other showing the attributes and values of vodka.

2.3.6 K-page

A k-page is a simple 2-column table that shows all of the knowledge associated with a concept. It is very similar to a frame but shows more information. In addition to attributes and values, it shows relationships to other concepts and less formal knowledge such as descriptive paragraphs of text, pictures and images. If created using a web tool, it includes hyperlinks to other concepts in the k-base and to other files outside the k-base (such as documents, image files and video files). Figure 4.12 (on page 108) shows an example of a k-page.

2.3.7 Advanced Modelling

We have seen in these sections on k-models that there are various ways of showing the relationships between pairs of concepts or the properties of concepts. Some projects require more advanced modelling. Let us take a quick look at three advanced modelling techniques.

2.3.7.1 Properties of Relationships

Sometimes we want to show that a relationship (triple) has properties. For example, suppose I have the relationship ‘Joe Smith – works for – Epistemics’. I may want to say how many years Joe Smith has worked for Epistemics. In this

case, I want an attribute ‘number of years’ associated with the relationship. Some k-base technologies can handle this and others cannot. If they can, we need a way of showing this. One way is to use a relationship matrix but instead of simple square symbols in the cells, we show the values of the attribute (as in Figure 4.5 on page 93). Another way would be to have a frame or k-page for the relationship ‘Joe Smith – works for – Epistemics’.

Properties of relationships are useful when modelling procedural knowledge. We saw earlier in Section 2.3.3 that a process map can include decision points that show the conditions under which the tasks that follow will occur. These conditions are usually shown as labels on the arrows from the decision point to the task. An example of this is on Figure 4.11 (page 107) where the labels ‘Yes’ and ‘No’ are used after the ‘JICs Required?’ node. To model this in the k-base requires an attribute of ‘condition’ on the ‘followed by’ relation, with values ‘Yes’ and ‘No’.

2.3.7.2 Relationships Between Relationships

On rare occasions we need to show a relationship between a concept and a relationship, or between two relationships. To do this, requires treating one or more relationships as concepts. This is called ‘reification’ which means making real. For example, ‘Joe Smith – works for – Epistemics’ could be reified as a concept so it can have relationships with concepts or other reified relationships. An example of this might be to model the fact that Joe Smith works for Epistemics in their Nottingham office.

2.3.7.3 Rules and How to Model Them

Rules are statements of the form ‘If... then...’. For example: ‘If the room is on fire then activate the fire alarm’, or ‘If the animal has wings and feathers then it is a bird’. These statements contain one or more concepts in complex combinations with other concepts or attributes and values. Hence, we require a complex form of modelling. One way is to split the antecedent (the part before ‘then’) and the consequent (the part after ‘then’). The consequent may already be a concept such as a task. If not, then we can model it using one of the techniques described above. Finally, the relationship between the antecedent and consequent can be a simple one such as ‘implies’. Note if we want to keep things simple, we can just enter the rule into a k-page as a piece of text rather than formally modelling it.

2.4 47-step Walkthrough

We have seen in the previous sections three important aspects of a KA project: knowledge capture, knowledge analysis and knowledge modelling. In particular, we have seen many ways of capturing and displaying knowledge. Let us now see how these elements fit together into a project. This will introduce you to each of the 47 steps in the procedure described in the next four chapters.

To do this I will take you on a very brisk walk through a typical project to show what happens at each stage. Details of each step and variations from the norm are described in detail in Chapters 3 to 6. During the walkthrough, I will assume you are the knowledge engineer conducting the project.

2.4.1 Phase 1: Start, Scope and Plan the Project

The starting point for any project is to identify a project idea. This includes how the project can benefit the organisation and what the project would involve (Step 1). Opinions on the project idea are gathered from all relevant people (Step 2) and the idea is then documented as a 'project proposal' (Step 3). Agreement is sought for the project proposal from all key people (Step 4).

The initial phase of knowledge capture can now begin in order to scope the project, *i.e.* define which specific areas of knowledge that will be acquired. To start this, you (the knowledge engineer) creates a k-base (Step 5) so you have a place to put the knowledge. With the involvement of domain experts, you break the domain into different topics (Step 6) and then rate or rank these topics against key criteria (Step 7). Using this information, you identify a proposed scope and finalise it with the relevant people (Step 8).

You can now identify the sources of knowledge that will be needed during the project (Step 9). It is a good idea to identify exactly what sort of project you are doing (Step 10) which will help to define and understand the procedure that you will follow for the remainder of the project (Step 11). The procedure is used to create a project schedule (Step 12). The final part of the first phase is to collate the project proposal, project scope and project schedule into a project plan and disseminate this, and any other materials, to the project team (Step 13).

2.4.2 Phase 2: Initial Capture and Modelling

The starting point for Phase 2 is to learn the basics of the domain from documents or informal conversations with domain experts (Step 14). Following this, you prepare for the first one or two semi-structured interviews (Step 15) and then conduct these interviews (Step 16). The audio-recordings of these interviews are transcribed, either fully or partially (Step 17).

Using the interview transcripts and domain documents, you analyse the knowledge to identify the concepts that will form the main structure of the k-base (Step 18). A concept tree can now be created to develop a taxonomy of the concepts (Step 19). The concept tree is validated with the domain experts and added to where necessary (Step 20).

At this point, it is useful to take stock of the project and update the plan if necessary in light of what you have learnt (Step 21).

A k-page describing each concept is created (Step 22), and a glossary of all domain-specific terms is begun (Step 23). A meta-model is defined that shows how the k-base will be structured in terms of the relationships between concepts and the properties of concepts (Step 24). The meta-model is used to set-up the structure of the k-base and to define the templates for k-models (Step 25).

The relationships between concepts are entered using the appropriate k-models (Step 26). The properties (attributes and values) of certain concepts can be entered if required (Step 27). Knowledge of processes and how the expert performs certain tasks is entered into the k-base (Step 28).

Phase 3 ends with one or more validation sessions with the experts. A number of k-models are used to validate that the knowledge is correct, complete and

relevant to the project (Step 29). A first-pass k-base is now in place, which for some small projects may be all that is required.

2.4.3 Phase 3: Detailed Capture and Modelling

Further interviews and modelling activities are used to capture more detailed knowledge (Step 30). The scale of the project and the requirements of the end-product will determine the extent of this step.

The main k-models can now be finalised (Step 31) in preparation for the creation of a prototype end-product (Step 32). This prototype is used to carry out an assessment exercise with a representative sample of end-users (Step 33). Based on the results of the assessment exercise, a completion plan is produced that defines the actions required to complete the project (Step 34).

In light of feedback from end-users and the project scope, the capture of very detailed (deep, tacit) knowledge now takes place using a suitable set of specialised techniques (Step 35). If required, cross-validation takes place with secondary experts checking that the k-models developed with the primary experts are correct and best practice (Step 36). Any differences of opinion between experts are identified, discussed and resolved at a consensus session (Step 37).

Before finalising the k-base, a check is made to ensure that all of the knowledge has been validated and its status has been recorded (Step 38). The k-models and the underlying k-base can now be finalised (Step 39).

2.4.4 Phase 4: Share the Store Knowledge

The final phase starts by defining and creating the format of the end-product (Step 40). Using the k-base and the format, a provisional end-product is created (Step 41). This provisional end-product is given a full assessment by end-users to identify modifications and improvements (Step 42). The final end-product is then created (Step 43) and released for use in the organisation using the appropriate release procedure (Step 44).

The end-product is publicised to the organisation so that all potential end-users know what it is, where it is and how it can be used (Step 45).

After the end-product has been used for some time, its impact on the organisation is assessed and documented (Step 46). Finally, a complete project review takes place to learn lessons and make suggestions that can be used to improve the methodology and support systems (Step 47).

2.5 Ensuring the End-product is Useful, Usable and Used

Let me bring this chapter to a close by describing how the different elements in the 47-step procedure contribute to three important aspects of the end-product: (i) End-users find it useful; (ii) End-users find it easy to use; (iii) End-users actually use it.

Table 2.1 shows how the steps contribute to these aspects of the end-product and the associated reasons.

Table 2.1. The impact of each step on the end-product with reasons

Steps	Impact on the end-product			Reason
	Useful	Useable	Used	
1-4	✓			Creates a good project proposal with input from all key people
5	✓	✓		Creates a k-base that is a central, structured store of knowledge
6-8	✓		✓	Defines a project scope by involving domain experts and end-users
9	✓			Identifies all sources of knowledge to be consulted during the project
10-13	✓			Creates a good project schedule that maximises the resources available
14-20	✓	✓		Uses effective methods for capturing, structuring and validating knowledge
21	✓			Reviews the project aims so they can be re-aligned for maximum benefit
22-30	✓	✓		Uses effective methods for eliciting, modelling and validating knowledge
31-33	✓	✓	✓	Creates a prototype end-product that is assessed with end-users
34	✓	✓		Reviews the project plan so tasks are focused on providing the best end-product
35	✓			Uses effective methods for eliciting deep, tacit knowledge
36-38	✓			Ensures that all of the k-base contents are correct, complete and best practice
39-43	✓	✓	✓	Assesses a provisional end-product with end-users and makes the required changes
44		✓	✓	Releases the end-product in the right way and provides information and training
45			✓	Publicises the end-product to all potential end-users
46	✓	✓	✓	Assesses the impact of the end-product and makes any necessary modifications



<http://www.springer.com/978-1-84628-860-9>

Knowledge Acquisition in Practice

A Step-by-step Guide

Milton, N.R.

2007, XII, 176 p., Hardcover

ISBN: 978-1-84628-860-9