Towards Continuous Knowledge Engineering

Klaas Schilstra and Pieter Spronck
Knowledge Based Systems Department
TNO, Netherlands Organisation for Applied Scientific Research
Delft, The Netherlands
{k.schilstra; p.spronck}@bouw.tno.nl
www.tno.nl

Abstract: Continuous Knowledge Engineering is an alternative approach to knowledge engineering that embraces the philosophy that knowledge systems are open-ended, dynamic artefacts that develop through a learning process in reaction to their environment. This approach is based on the debate on symbolic and situated models of human cognition and aims to make knowledge engineering a learning process by adherence to incremental development, participation of the expert and direct application of the knowledge system. This paper presents this approach to knowledge engineering, examines a knowledge modelling tool that supports it and looks at an application of this tool.

1. Introduction

There has been extensive discussion on the symbolic or *descriptive* stance versus the situated theory of human cognition [1,2]. This discussion has opened up a re-evaluation of the physical symbol system hypothesis that has been held as defining for knowledge system development and knowledge engineering from its very conception. The hypothesis defines a theory of cognition based on explicit representation of knowledge and the goal-oriented or rational use of this knowledge by agents [3].

The situated cognition hypothesis instead proposes that all knowledge is grounded and connected to the context in which it is acquired and applied. This would imply that as knowledge is used it will change with each new experience and that static representations of knowledge degrade. This questions many of the assumptions that underlie knowledge engineering today. The proponents of *strong situated cognition* profess that only reactive systems, based on learning and inductive approaches, devoid of representation and reasoning would therefore have sufficient and necessary facilities for intelligent action [4, 5].

A middle way is *weak situated cognition* [1], which says that as knowledge changes through experience, an approach that permits a knowledge model to follow that change may circumvent the adverse effects. In addition, this would allow knowledge engineers (KE) to use current approaches but with a change in perspective on the employment of those techniques. In short, knowledge

engineering must be a learning process, bound to the practical use of the system. This would seem to be a natural progression from the current model-based approaches towards an approach that embraces change in a knowledge model through learning.

This paper will first give a short overview of currently experienced problems and aspires to provide some explanations for these problems. In addition a knowledge engineering approach based on a learning process is proposed that aims to circumvent or solve these problems. A tool is described that supports this approach to knowledge engineering and an application of the tool is discussed. Finally some conclusions are reached and some future work is discussed.

2. Problems in Knowledge Engineering

The state-of-the-art in knowledge engineering is to a great degree a result of the symbolic view of knowledge and cognition. It offers a constructive model of human reasoning as operations on symbol patterns. In this way it has been a successful cognitive model leading to useful systems. In concordance with the statements by Clancey [2], these accomplishments should not be detracted from. From the earliest systems to the systems developed using mature methodologies, such as CommonKADS [6], certain problems are associated with the development of knowledge systems.

From a practical and commercial perspective the most important problem is the cost and risk associated with knowledge system development. This makes it difficult to 'sell' knowledge systems as viable solutions to problems that organisations may experience, and creates a situation where knowledge systems are still perceived as experimental technology. Not many companies allow mission critical applications to be based on knowledge technology as yet, although there are some notable exceptions, for example XCON, which supports DEC's configuration management and has done so for many years [7]. This system has further displayed considerable benefits, saving the company millions a year in economic terms. But despite these successes and many others, war stories from the industry often relate of projects that do not rise above initial prototypes or unsuccessful introductions of finalized systems.

The cost and risk can be attributed mainly to the intrinsic difficulties of developing knowledge systems. The two main culprits are the knowledge acquisition and maintenance problem and the gap between prototype systems and industrial strength knowledge systems [8].

The fact that knowledge acquisition is hard is well known. The knowledge acquisition bottleneck is caused by the inherent difficulties in formulating and formalising knowledge, intensified by communication problems between the KE and the expert [9]. The expert and KE have to 'learn' to work with each other and create a combined view of the domain, to be able to structure the domain knowledge, identify important concepts, relationships, rules and heuristics.

After the development of a knowledge model and system, maintenance of knowledge models to remove flaws and to complete missing features is performed.

This is not commonly perceived as the natural degrading of knowledge or the shift in meaning caused by its situated nature. However, knowledge systems show considerable change over their lifetime. XCON shows that 50% of the rules may be modified, deleted or added during a given year [7], which can be explained as resulting from new products or configurations. As was shown more recently in the PIERS system, knowledge changes considerably even when the domain itself is static [10, 11] (see also below). The maintenance of knowledge system therefore contains a clear and sizeable component of knowledge acquisition and modelling.

The gap between a successful knowledge model and the development of a user-friendly, industrial strength knowledge system constitutes a further problem. Even after a successful knowledge model has been developed, the ability to translate this model into an application acceptable to the end user is tenuous. Besides misconceptions and neo-Luddite fears, the users often find the level of such applications to be below par. The effort to develop user-interfaces of a quality that most users have grown accustomed to and inclusion of auxiliary facilities, such as integration with existing information systems, can require even more work than the total effort of developing the knowledge model itself. This is discussed in literature [8] but also has been observed in applications developed by the authors. The percentages seem to show that this additional development constitutes from 50% to 80% of the total development effort.

Current research approaches aim to address the knowledge acquisition and maintenance problems mentioned above by increasing the control and management of knowledge engineering, usage of deep models and reuse of existing knowledge models. Problems with the development of actual applications are seen as a tooling problem. A prime example of this is CommonKADS [6], which places much importance on methodology and constructing the right model, and implementing it. The model library of CommonKADS is extensive and allows for reuse of both problem solving methods (PSMs) and domain models. Knowledge models are developed using a risk-driven spiral development process. The goal is to develop a model that goes towards a final system that can be maintained for flaws and incompleteness in its functionality. CommonKADS further attempts to moderate the problems in development of the final system by giving due attention to the organisation, the technical environment and modelling the communication with the user.

Knowledge system development embracing situated knowledge acquisition is also performed but is much smaller in scale and has yet to be embraced by the mainstream. The results from the PIERS system are exemplary for the benefits that can be gained by using incremental approaches based on grounding the system in its environment [10, 11]. It is also one of the few examples where experts or domain specialists are actively involved in the knowledge modelling process using ripple-down rules (RDR). The system has been developed without any design or model, by the active involvement of the experts and an incremental prototyping approach. The maintenance proceeded on a case to case basis. By spending on average a couple of minutes a day the system was maintained. Every change is

immediately part of the operational knowledge system. Using this approach the system operated with considerable success, reaching accuracy levels over 95%.

Considering knowledge acquisition as a process of learning, can explain the inherent difficulty of knowledge acquisition as well as the inordinate amount of maintenance required. If knowledge originates in experience and is intimately tied to the environment in which it is acquired and employed, then this would question the current research approaches. It would place doubt with design-based approaches that would try to create a description of the knowledge and the domain needed to perform within that domain without grounding them in experience. Another of the consequences of accepting the weak situated cognition position is that we can not expect to reuse existing models or PSMs *as is* to bolster the productivity or share the cost of development of a knowledge base. The problems that are experienced in creating knowledge models and the results from the PIERS system seem to indicate that there is at least a place for improvement. If the situated view is taken the focus should be on how we build and change models, to allow room for continuous evolution and exploration, characteristic of any learning process.

Coming back to the earlier comments on the ability to sell knowledge technology as a viable solution, for the art of knowledge engineering similar remarks can be made. If research and development in knowledge engineering can improve the applicability of knowledge systems, and increase the number of experiences in the form of successful and unsuccessful knowledge systems, the volume of these experiences will allow the improvement of the best knowledge of knowledge engineering, as well as further the cognitive theories these systems embody.

3. Continuous Knowledge Engineering

The Continuous Knowledge Engineering (CKE) approach embraces the philosophy that knowledge systems are dynamic, open-ended artefacts that must develop in response to their environment. Therefore, developing knowledge systems requires that a development methodology is adopted that is based on a learning process. Learning is seen as a continuous process of incremental insight, change and application leading to further insights.

The CKE approach is not so much a methodology itself but a change of perspective on the deployment of current methodologies, tools and techniques; a set of ideas that allows a learning process. The approach centres on three principles to accomplish this: incremental development, active participation from non-specialists such as experts, and direct application of the knowledge system. Each of these principles will be discussed in the sections below.

3.1 Incremental Development

Incremental development means developing a following step based on the previous one. Basically, lessons learned in the current iteration can be applied to better enable and inform the tasks in next ones. This is the reason why many evolutionary

or incremental software engineering methodologies favour it to enable risk- or functionality driven approaches [12, 13]. As was discussed in the previous there is already a tendency to employ iterative strategies in other approaches to knowledge engineering. But in these approaches the iteration mainly takes place in the design phases, and not through cycles of implementation, application and design.

In many situations a solution to a problem can not be created, but must be revealed through experimentation and exploration. Only when the knowledge is made explicit can certain epiphanies take place, clarifying and illuminating old problems in a new light. And only by positioning the knowledge models in the real world, where they are confronted with real problems and real solutions can we be assured of the quality of that model and gain experiences leading to novel insights. Anything less than this may provide a candidate with which we can delude ourselves. This harks back to the grounded program defined by Brooks [14].

The ability to incrementally develop and ripen a knowledge model, to incorporate new knowledge and improve on the knowledge already present in the system is the pivotal element in the approach. Knowledge builds on knowledge, and true insights can only be gained working and testing our incomplete models of the world and discovering imperfections in its understandings. The focus of the CKE approach is therefore on fast and frequent delivery of the knowledge system. In this way the CKE approach becomes based on the experiences with the product, rather than on detached requirements or models. This is more flexible as the system does not have to constitute a complete solution, as long as it demonstrate evolution and can be used a basis for evaluation of the progress and direction of the development.

3.2 Active Participation

The added insight gained by being involved building an explicit knowledge model is placed with the wrong person, when it is performed by the KE. The person best able to appreciate the possible insights to be gained from the development and application of the knowledge model is the expert. The practical experience of the expert should give the expert a better basis to learn and make new discoveries. The expert is grounded in the field and can bring his or her own practical experiences to bear. The expert will also gain the most benefit from interacting with the model. Making their understanding of the task and the domain explicit will aid experts in deepening their own knowledge.

The KE should focus on the structuring of the knowledge system and facilitating the development of the system. Furthermore, the iterative way of working would require the KE to be either constantly connected to this single project. This could be done if the effort was limited but regular or when the knowledge model would be large enough to warrant such singular attention. But this would not be very effective or efficient and would most likely also not be appealing to such persons.

Active participation focuses on getting the domain specialist to be actively involved in modelling knowledge. This can be in unison with a KE, but the ultimate intent is to get the expert working directly on the knowledge model. The wish to have the expert participate in the knowledge engineering has existed for long as it represents

certain financial and practical realities. However, knowledge modelling by the expert is not a common approach in AI, although examples are known (e.g. RDR [10], Spark-Burn-Fire Fighter [11]). From these examples it is clear that it is possible, but there is a considerable need for support from the tools used to create these systems.

3.3 Direct Application

Direct application means that the knowledge model can be deployed to its intended users early in the development and after that with each increment. Simply said, this means that a system based on a single rule should be both consultable and deployable with ease, and that changes to the knowledge model should also be published in rapid succession. Furthermore, the quality of the systems should be such that it is acceptable to its users.

One of the main priorities in development of knowledge systems is that someone will use the system – it has, at least, to meet with interface standards that end-users are expecting from traditional application programs [8]. The fitness of the knowledge system to the expectations of the users is an essential criterion for the acceptance of the knowledge system and the knowledge it contains. This kind of focus on 'business purpose' can also be seen in other incremental or evolutionary development methodologies such as DSDM [13]. There is no reward for delivering the best product, just for the delivering a good solution.

As was mentioned considerable effort goes into developing the visualisation and auxiliary facilities for a knowledge systems as well as integrating the system with existing information systems or as a component of a larger system. Making this easier or spreading this effort over a longer period, may help easing this and discover the best form in interaction with the users and the organisation.

3.4 Conclusions

On the surface, CKE may seem to propose a return to design-less, rapid prototyping approaches, and 'to do away' with mature methodologies such as CommonKADS. Rather, CKE proposes a change in perspective, from model-based to learning-based knowledge acquisition and knowledge system development, that can be applied to any development methodology. It further aims to direct the development of tools that can support the use of a methodology that supports CKE.

The realisation of the three principles places the development of knowledge systems under considerable stress as it requires the existence of tools and infrastructure that can support the frequent iterations, participation of the expert and recurrent deployment of the knowledge system as well as the development user-interfaces and auxiliary facilities. In this technical support is essential to make these CKE approach viable and practical. An investigation into the features that such technical support must embody is described in the remainder of this paper.

The intent of the tool support and the applications described below is to show how, using the technologies available today, the practice of knowledge engineering to can be made to embrace the dynamic and open-ended nature of knowledge and enable a learning process. This focuses initially on enabling participation of domain experts and improving the quality of the applications developed with these knowledge models.

4. A Simple Tool

The Knowledge Base Editor (KBE) is a knowledge modelling tool based on a philosophy of simplicity. It does not have an elaborated domain-model and its inference mechanism is based on simple backward chaining. The basic technology contained in the system is therefore not an elaboration in a technical sense, rather it can be perceived as a simplification of the technical possibilities that can be discerned in many other tools. This simple core is coupled with visual representation of the knowledge in the form of decision tables and extensive facilities to deploy the knowledge models. Furthermore, these deployment facilities are highly configurable and can be used to quickly and easily develop customised and dedicated user interfaces.

4.1 Key Features

The KBE has a straightforward knowledge system architecture. The knowledge is contained by a knowledge base component consisting of a domain model, a knowledge model and inference mechanism. Each of these will be described in the sections below. The difference with most other systems is that the core of the system is a component that does not contain any visualisation aspects, much like a database. The visualisation is seen as separate issue, allowing development of many different visualisations and media. This allows beyond the development of dedicated domain specific applications, including the construction of domain specific knowledge model editors. In this sense the KBE is only a default interface placed on top of this component, to allow knowledge models to be edited and maintained. The component-based approach further allows easy integration of this knowledge technology as part of a larger system. As such knowledge systems have been integrated into document information systems and GIS systems.

4.1.1 Primitive Domain Model

The domain model is formed by a set of parameters. A parameter is basically an attribute-value pair, where the value can be one of four base-types: String, Integer, Real, and Boolean. String parameters can be given a domain, which enables it to operate as an enumerated value. Boolean parameters can be configured as to what term is used to denote true and false. Each parameter further has a prompt, an explanation, a domain and a default value. The prompt and explanation are used to query the user for a value when the inference can not yield a value for the parameter. The domain can restrict the possible values that a parameter can accept or state the possible enumerated alternatives that can be chosen for the parameter.

4.1.2 Visual Knowledge Representation

	Wine type	R1	R2	R3	R4	R5
C1	Main course	red meat	poultry		fish	ELSE
C2	Turkey is served	-	yes	ELSE	-	-
A1	Wine type	red	red	white	white	unknown

Figure 1 An example of a decision table

The knowledge model consists of a set of decision tables. The decision table allows visual representation of a number of related rules. The decision table has been around for some time and has been used for knowledge representation in other systems as well [16]. The left top part of the table contains the conditions, below which the actions are located. These contain references to parameters in the domain. The right side contains a tree of condition-alternatives and action-alternatives. The condition alternatives and action alternatives can contain expressions yielding a value. This can be simply a value in the domain of the parameter or a reference to another parameter. Special values include the don't-care (-), which means that the condition is not relevant and can be ignored, and the ELSE alternative, meaning the remainder of the domain.

The condition parameter is compared to the active alternatives, for example *Main course* is compare the values *red meat*, *poultry*, *fish*. The chosen value is used to determine the next set of alternatives, below the chosen alternative, as visible in the figure. If there is a don't care, the condition is skipped and the condition alternatives in the row below are chosen to be executed. When all conditions have been evaluated a column of the table's action alternatives is selected, and these actions are performed, in this case assigning a value *red* to *Wine type*. By virtue of this visual representation, it is easy to examine the contents as to their completeness and exhaustiveness. Therefore it is easy to decide whether one of the required alternatives is missing or that there are no actions associated with a particular configuration. Furthermore it is easy to add additional knowledge at any point in the decision table, as an additional condition, condition alternative, or the consequences of the table's execution can be extended with additional actions.

Some additional fineries exist but these are limited compared to this main mechanism. Backward chaining inference is easy enough to be explained to an expert and can be made more clear by showing an example consultation. The simplicity of the parameters and the visual nature of the knowledge representation allow an expert to comprehend the effect of a change to a parameter or decision table and the resulting inference well enough. In limited experiments experts have been shown to be able to enter knowledge into the system independently or with limited support from the KE.

4.2 Modelling and Application Development

Knowledge modelling can take many forms, but is mostly an incremental process that can proceed top-down or bottom-up, or using a mixed approach. The most important feature is that the minimal system that can be consulted is very small. A single decision table and a couple of parameters are sufficient for a first implementation and simple knowledge systems require no more than this. The knowledge model can be consulted at any time. This allows testing of the knowledge model to its ability to deal with a specific case. As each parameter can be used a goal, testing parts of the knowledge system, up to an individual decision table is easy. A decision table can therefore be developed incrementally as a separate functional unit. Cases can also be used to perform more general tests.

There are several ways that a knowledge model can be deployed. Standardised solutions include a stand-alone system, and a web server extension that can dish out HTML pages. Both systems operate on a series of default HTML templates, but it is possible to define a custom page for a parameter and to create elaborate pages that answer several queries at once. Dedicated systems can also be developed using knowledge-aware components that are used to create sophisticated user interfaces within standard software development environments. An example of this will be shown in the application below.

What the description shows is a simple knowledge system development environment. It is easy to understand and a knowledge model can be comprehended to its dynamic behaviour with little knowledge of information technology, i.e. usable by experts. The development of a knowledge system can be performed incrementally with ease. Furthermore it has the ability to consult the knowledge base at any moment in time, which supports verification and validation. In addition, deployment can be reasonably simple using the technologies described earlier and facilities for building dedicated systems are integral to the system's architecture. In these ways the system supports the three principles of the CKE approach.

5. Building Materials Regulation Knowledge System

The building materials regulation knowledge system (BOKS) is a knowledge based system that implements the Dutch regulations on the use of building materials[1] (BSB). BOKS is also described in more detail in another paper [17]. The BSB is a complex set of rules, which many people need to adhere to. A knowledge system was seen as an ideal way to allow people to check whether they are using building materials in conformance to the law. It is currently used by approx. 3000 people in the Netherlands, constituting a large majority of those who need to work with the BSB.

---

[1] The Dutch term is 'bouwstoffenbesluit', hence the abbreviation BSB. More information on the legislation can be found in [18] for those with an appreciation of the Dutch language and a fondness for legalese.

BOKS is a stand-alone computer program that contains knowledge about the BSB and is an example of a dedicated system developed using the KBE. BOKS can answer questions posed by the user, by using this knowledge and by asking questions to the user such as “what is the type of material”, “where is it used” and “what are the emission values the laboratory reports”. After such a consultation it gives a report that conforms to the same conclusions an expert would reach.

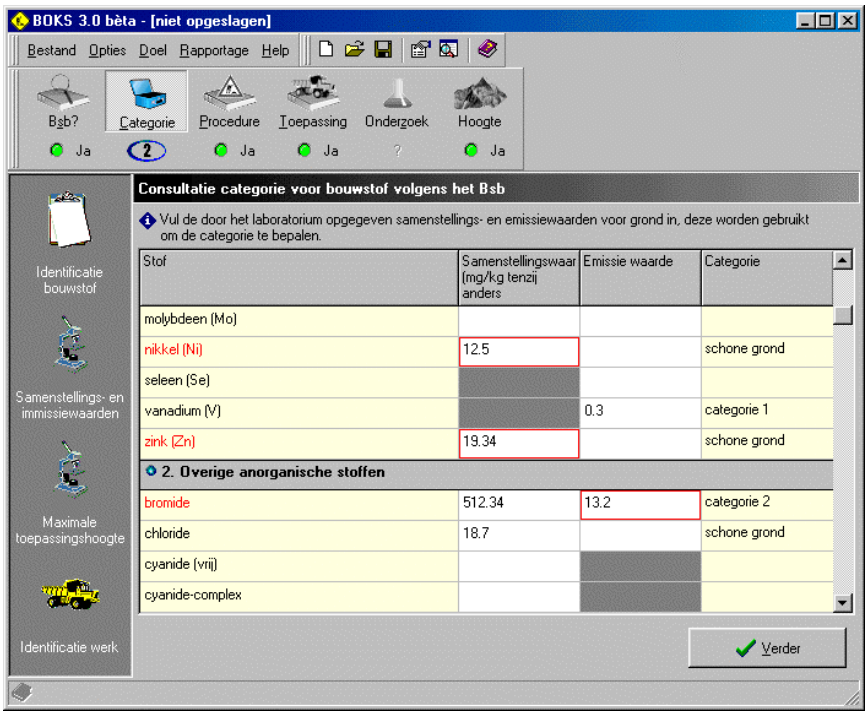

Figure 2 BOKS User Interface

The nature of all the questions asked by the system is such that a layman can answer them. The system can be consulted to answer six goal questions:

- Is the BSB applicable to the current situation,
- Which is the category the materials belong to,
- What procedures must be followed when the material is used,
- What requirements are posed on the application of the material,
- How should a sample of the material be taken and examined,
- How high is the user allowed to stack the material?

These questions offer enough information to allow a user to know they adhere to the BSB. The user is provided with an extensive report of the conclusions and reasons behind the results, and a list of the answers given to the system.

The advantages of using the system are similar to those of other knowledge systems. The benefits of BOKS compared to a human expert, make the knowledge system to be preferable in most respects, especially since there are very few human

experts available. In this case the main benefit is the large number of users that would otherwise require training to acquire an intimate knowledge of the BSB, which they do not require now. It therefore negates the training of laymen in complex legislature, which they would have a hard time keeping up with, as modifications to the law are still being made. And as further proof of the effectiveness of the system, it is also being employed by the experts that have been involved in its development, because BOKS is quicker and more accurate in its findings than they themselves are.

5.1 Modelling Approach

The government publication of the BSB consists of 300 pages of text that is difficult to apply not only because of its sheer size, but also by the numerous tables, references, schemas, footnotes and exceptions. All of these have to be taken into account, and the knowledge for one specific aspect can be found spread throughout the document. Practice has shown that even experts find it difficult to use it correctly. On the other hand, once the BSB has been applied to particular situation, the results are easy to understand.

BOKS now contains about 180 decision tables and 220 parameters. During the development of the first two versions that were deployed and a third one that is currently being developed, these numbers have changed considerably (initially it contained some 120 tables), as legislation was changed or other modifications were necessary. The knowledge modelling was performed solely with the KBE. Based on the functionality of the different goals questions mentioned earlier, the system was developed by incrementally adding each of the six goals. Each goal functions as a sub-goal for other goals, therefore the development of the knowledge model in these subdivisions worked especially well.

The KE took the role of the intermediary, who would interview and discuss the global and specific parts of the BSB articles of law. Initially this was fed back to the expert as text notes. This practice was abandoned early on for an alternative propose and revise approach, where knowledge was entered by the KE in the system, and the decision tables and parameters were reviewed by the expert. This was used almost exclusively for the duration of the knowledge acquisition. This review allowed the KE to explain the content of the decision tables and perform some test cases with the system. In many cases these discussion would concern a single table, but two complete reviews of the knowledge as a whole were also performed.

The visualisation of the knowledge as decision table and the simple domain model and inference mechanism allowed the expert to not only understand it enough to validate and verify the modelled knowledge, but also to comment directly on the table, giving instructions to add an alternative or change a value in one of the cells. Sometimes this could extend to the suggestion to add a new table at a certain place. The ability to consult the system at each point in time also enabled some experimentation and the use of test cases. In other cases a single or a small set of decision tables would come under scrutiny by consulting these few tables.

Further results were that by posing an initial decision table, one could come to further clarity through iterating over the knowledge. An initial flawed formulation would enable reformulation and exploration of the actual knowledge representation that would best fit the sometimes difficult law articles. This has also led to a situation where the experts would many times faulted by their own knowledge. The consistent and complete representation of the decision table would contain a missing action alternative, indicating that a specific situation was not covered by the law. After the initial surprise the legislation would then be examined by the expert, who would try to come up with an answer.

From consultation of the system on test- and real life cases, using both release candidates and deployed versions, different errors and misconceptions surfaced. These were fed back into the model. In some cases it became obvious that these errors were caused by inconsistencies in the law itself, which have been reported to the legislative body. Furthermore, changes were made to the law in some areas. In effect the knowledge model was seen to grow considerably and the majority of the decision tables in the model have seen modification during their lifetime. The development of the system will remain to be an ongoing effort as more changes are made to the law and the system is applied to many more real world problems.

5.2 Development

The BOKS is a system that integrates different additional components. The knowledge component is only a part of the larger system; the application also contains a database of building materials, from which many of the questions otherwise posed to the user are answered, and integrates an XML-based reporting facility. The domain experts have maintained both the report templates and the content of the database.

The user interface developed for BOKS is made to be user-friendly and easy to use. Because of the simple questions it answers and the visualisation it supplies, people often suppose BOKS is no more than a friendly kind of spreadsheet. While this is the intended effect, this obscures the fact that it contains a vast array of knowledge, which it applied diligently to come to its final conclusions.

5.3 Conclusions

The application shows that the domain experts can understand the knowledge system and gain important insights into their own knowledge and, beyond this, the expert can make constructive comments directly on the knowledge represented in the system. The development of the model through small increments with the participation of the expert is perceived as critical in this respect. This gradual approach also aids the communication between the KE and the expert and makes the expert assert ownership of the model.

As to the ability for the expert involvement in the actual knowledge modelling, a final conclusion that was reached is that the expert would be able to develop parts of the knowledge model by themselves, of the granularity of a single decision table.

A problem that was discerned was the lack of structure in the domain model as a set of parameters, making it difficult to form a conception of the organisation and interconnections in the model. Locating parts of the model that either would need to be modified or that would be affected by a change in the model was made harder by this as the model grew. Some separate experiments confirm this result.

It further shows how from an initial implementation of a knowledge model, an iterative strategy can explore the knowledge, by experimentation and evolution. This improves not only the knowledge model, but also leads to significant insights for the domain expert both from the added clarity of an explicit model, revelations during modelling and from the application of the system.

6. Conclusions, Recommendations and Future Work

This paper has described an alternative approach to knowledge engineering that embraces the philosophy that knowledge systems are open-ended, dynamic artefacts that develop through a learning process in reaction to their environment. This approach is based on the debate on symbolic and situated models of human cognition and aims to make knowledge engineering a learning process by adherence to an incremental development, participation of the domain specialist and direct application of the knowledge system. This paper has presented this approach to knowledge engineering, examined a knowledge modelling tool that supports it and looked at an application of this tool.

The application and the tool make clear that the principles on which the CKE approach is based are at least attainable. The tools and techniques employed to this end are not drastically different from those employed in other knowledge system projects. This is in contrast with the RDR approach, which employs technology specifically engineered for that purpose. The application itself also shows the CKE approach is able to come to practical, industrial strength knowledge systems. Additionally, some evidence is given that the benefits that CKE proposes are also realised. The experts rather than the KE gains access to the knowledge model, and are able to improve their understanding of the domain. The technology used allows faster development of an initial system, which binds the system to its application, allowing experiences from application to feedback to the knowledge modelling. The iterative nature of the development allows gradual development of the knowledge in response to this improved understanding. This is perceived as lessening the knowledge acquisition and maintenance bottleneck.

The KBE is employed in different practical projects and further experiments with experts involved in modelling the knowledge with the tool are planned. Further development of the tool examines the role that object oriented modelling can play in creating a conceptual map of the domain, and to partition and organise the knowledge. This should alleviate most of the difficulties in locating the parts that need to be modified. It is thought that the object orientation will provide further assistance in making the models easier to build and change, through inheritance and encapsulation. This development is currently underway.

References

[1] Menzies, T.J. (1998). Towards Situated Knowledge Acquisition. In: Special Issue of the International Journal of Human Computer Studies “The Challenge of Situated Cognition for Symbolic Knowledge Based Systems”, pp. 867–893.

[2] Clancey, W.J. (1999). Situated Cognition: On Human Knowledge and Computer Representations. Cambridge: Cambridge University Press.

[3] Newell, A. and Simon, H. (1976). Computer Science as Empirical Enquiry: Symbols and Search. Communications of the ACM 19(3).

[4] McDermott, J. (1989). A Critique of Pure Reason. Computational Intelligence 3, pp. 151–160.

[5] Brooks, R. (1991). Intelligence without Reason. In: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91), pp. 569–595, San Francisco, CA: Morgan Kaufmann.

[6] Schreiber, G., Akkermans, H., Anjewierden, A., et al. (1999). Knowledge Engineering and Management. Cambridge: MIT Press.

[7] Bachant, J. and McDermott, J. (1984). R1 revisited: Four years in the trenches. AI Magazine 5(3), pp. 21–32.

[8] Crofts, A.E., Ciesielski, V.B., Molesworth, M., Smith, T.J., and Lee, R.Y. (1989). Bridging the Gap Between Prototype and Commercial Expert Systems – A Case Study. In: Applications of Expert Systems, J.R. Quinlan (ed.), Reading, MA: Addison-Wesley, pp. 93–105.

[9] Buchanon, B., Barstow, D., Bechtel, R. et al. (1983). Constructing an Expert System. In: Building Expert Systems, Hayes-Roth, F., Waterman, D., Lenat, D. (eds.), Reading, MA: Addison-Wesley.

[10] Compton, P., Horn, K., Quinlan, J.R., and Lazarus, L. (1989). Maintaining an Expert System. In: Applications of Expert Systems, J.R. Quinlan (ed.), Reading, MA: Addison-Wesley, pp. 366–384.

[11] Compton, P. and Jansen, R. (1990). A Philosophical Basis for Knowledge Acquisition. Knowledge Acquisition 2, pp. 241–257.

[12] Jacobson, I., Booch, G. and Rumbaugh, J. (1998). The Unified Software Development Process. Reading, MA: Addison-Wesley.

[13] Stapleton, J. (1997). DSDM: Dynamic Systems Development Method. Reading, MA: Addison-Wesley.

[14] Brooks, R. (1991). Intelligence without Representation Artificial Intelligence 47, pp. 139–159.

[15] Marques, D., Deallemagne, G., Klinker, G., McDermott, J. and Tung, D. (1992). Easy Programming: Empowering People to Build Their Own Applications. IEEE Expert June, pp. 16–29.

[16] Montelbano, M. (1973). Decision Tables. Palo Alto, CA: Science Research Associates, Inc.

[17] Spronck, P. and Schilstra, K. (2000). BOKS: A Rule-base System in Support of the Dutch Building Materials Regulations. In: Proceedings of the Sixth Pacific Rim International Conference on AI (to be published).

[18] Dutch Ministry of VROM (1998). Bouwstoffenbesluit, Parts 1, 2, and 3.