Project Management and the Evolutionary Development of Expert Systems

Ian Watson

Department of Surveying, University of Salford,
SALFORD, M5 4WT
E-mail I.D.WATSON@CONSTRUCT-IT.SALFORD.AC.UK

Abstract

The development of expert systems is inherently uncertain and so involves a high degree of risk. This paper describes a project management method that helps manage this uncertainty. It has been tailored to the Client Centred Approach — an expert system development method that is being designed for use by small and medium sized enterprises. This context implies that the management technique and its accompanying documentation must not over burden the resources of a smaller developer. The helix method of project management introduced in this paper represents a different view of Boehm's Spiral Model. It accepts that conventional linear project planning methods are not always suitable for developers of expert systems. Having accepted this, the helix method allows plans to be made for each development stage within the Client Centred Approach. We believe the Client Centred Approach is applicable wherever prototyping is used, and we contrast it with the methods being developed by KADS-II.

1. INTRODUCTION

This paper describes proposals for handling project management within the Client Centred Approach (CCA). The principles of the CCA are described in Watson et al., [1992a & b] The thinking behind the approach, and its current state of development, are described in greater detail in Basden *et al*. [1995].Although the technique described here is applicable for any project that uses prototyping (or RAD Rapid Application Development) to develop a system (around forty five per cent of all commercial expert system projects according to a survey [DTI, 1992]), it has been developed specifically for small and medium sized enterprises (SMEs), rather than larger organisations. The DTI's report showed that forty per cent of expert system (ES) applications are being developed by SMEs (*i.e.*, companies with less than five hundred employees).

While ESs remained within the research labs as largely experimental demonstrators, there was less necessity to manage their development. That is, an ES would take as long to develop as the research grant provided for, or its development would last until a doctoral thesis was submitted [Inder & Filby, 1991]. However, as the development of

In, Research & Development In Expert Systems IX, Bramer, M. & Milne, R.W., (Eds.), pp.61-73. Cambridge University Press, 1992.

ESs has become more routine, insofar as many are now being developed by and for commercial companies, project management is becoming a central area of concern [Bright *et al*., 1991; Klahr, 1991; Taylor *et al*., 1991; Thomas, 1991].

For conventional information systems development (*e.g.*, databases), the main determiners of the size of the eventual system are the number of data items, functions performed on them, and input/output routines of the desired system. The quantities of each of these can be estimated reasonably accurately at the start of the project. From these estimates, development time scales can be calculated from experience. However, with ESs this is rarely possible [Thomas, 1991], although attempts are being made to evaluate metrics for ES development [Moores & Edwards, 1992].

Expert systems contain human knowledge that is used for problem solving. This expertise is the main determinant of the project size and complexity. However, there is no accurate way of estimating the effort involved in obtaining, structuring and representing this knowledge until a substantial amount of work has been done [Thomas, 1991]. Project planning must therefore be flexible, since early plans will not be accurate. The managers of expert system projects therefore need a project management technique that controls the flexibility while maintaining the visibility and accountability for all aspects of the project.

The CCA addresses this by making the development of an ES more visible through a well-defined project management method that explicitly deals with threats; *i.e.*, areas of uncertainty or risks that may jeopardise the project.

This paper first outlines the CCA's background and stages. It then describes a different perspective (or view) of Boehm's Spiral Method. The types of documentation that should accompany the evolving system are described, along with the project management activities. The paper concludes by outlining the potential benefits of the CCA's project management method to ES developers.

2. THE CLIENT CENTRED APPROACH

2.1. The Background

Basden [1989] argues that a problem common to most current ES development methods is that they are technology centred. They place too much emphasis on the activities used to develop the systems, such as "elicitation," "implementation," and "verification," and not enough emphasis on what the clients can see and understand. It has been argued that by putting people at the centre of the development process [Diaper, 1987 & 1989] there is a greater chance of the resulting system being useful. The DTI advises that *"involving the users in all aspects of the system development from the outset will help to avoid potential problems"* [DTI, 1992].

Basden, however, identifies the *"client"* as an individual or group distinct from the eventual end users of the system. Thus, in a corporate environment the client may be the senior management commissioning the system and not the eventual end users. The CCA covers the full development life cycle of an ES providing milestones to guide the project. These milestones refer to what the clients can see being demonstrated and not to the conventional tasks such as elicitation, acquisition, representation. This accepts that the clients may not understand the jargon or the distinction between the tasks involved in development but will be able to perceive demonstrable changes in the system.

2.2. An Overview of the CCA

The stages of the CCA are illustrated in Figure 1 and are described in more detail below. The CCA is divided into two broad activities:

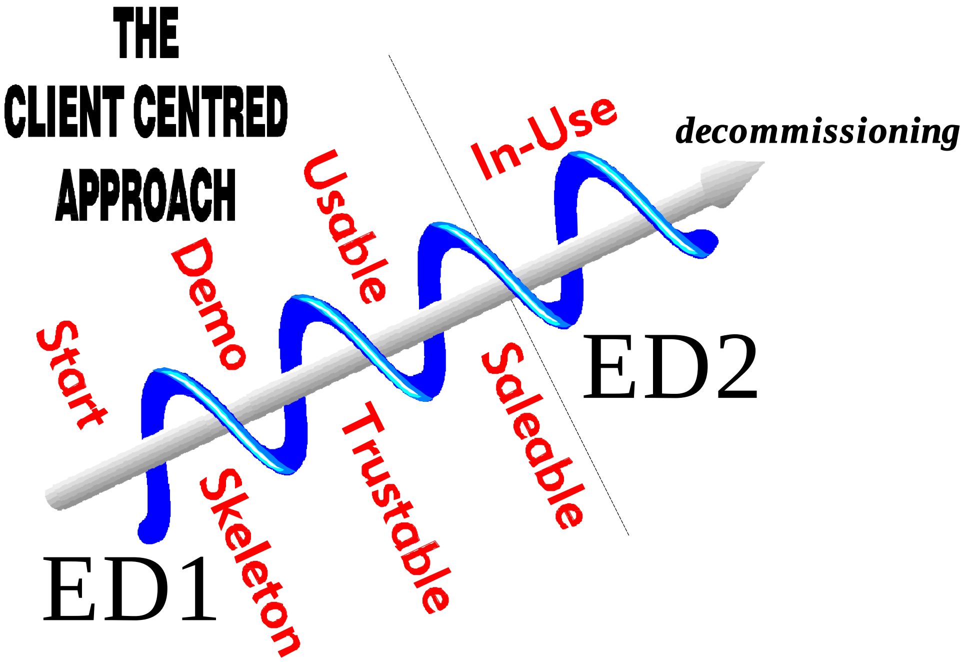

Figure 1. Seven Stages of the CCA

- **Evolutionary development part one (ED1).** This considers the development of the ES and takes it to a saleable stage. Each stage is a deliverable that the client can see, and that the developers must meet and plan for.

- **Evolutionary development part two (ED2).** This considers how the system can be kept in regular beneficial use, and considers such factors as training of users and most importantly the maintenance of the system. This phase only ends when the ES is decommissioned.

- *Start (the 5 hurdles)*

  The purpose, roles, benefits, and stakeholders are identified. The impact of the system on the client organisation is discussed, and those involved with the project get to know each other. As part of this process developers should considering five crucial questions or hurdles:

  1. Is the problem suitable for computerisation?
  2. Is the problem suitable for expert systems?
  3. Is the knowledge available to solve the problem?
  4. Is the system worth developing?
  5. Will the system be used?

  An ES is considered appropriate only if *all* the hurdles are crossed. The deliverables are documents outlining the feasibility of the project giving a "holistic picture" of the project.

- *Skeleton System*

  The deliverable is a mock-up that behaves and looks as the final system might but contains little knowledge. It is simply a set of interactive screens that show a few dummy questions, provide some dummy examples and possibly a report. The purpose of the Skeleton System is to let the clients see what the system might eventually be able to do, and how it might do it. It is also a vehicle for discussing the form of the inputs and outputs of the system, and is therefore a tool for knowledge elicitation. It can also be used to explore user interface requirements and other aspects of system functionality, similar to Colebourne *et al*. [1992]

- *Demonstration Systems*

  During this stage and the following stages iterative cycles of prototyping occur. Therefore, there may be several demonstration systems, each demonstrating a different aspect of the system's functionality. The first prototypes contain real domain knowledge, but can only produce acceptable results in a limited subset of the domain. Nonetheless, they demonstrate to the client that the system can solve the problem or let the project to be re-evaluated if necessary. This stage is used to explore issues relating to knowledge representation and system architecture before committing to a particular approach. The deliverables are the demonstration systems.

- *Trustable System*

  The knowledge in this deliverable is complete and correct. It gives correct results to all the problems the system will encounter. However, it will be difficult to use (even by its creators) and will be prone to operational problems.

- *Usable System*

  This deliverable has a usable interface and can link to external software if necessary. It also provides useful explanations, “what-if” facilities, and reports. This version could be used for real business benefit by those sympathetic to the system. Meeting this deliverable should involve evaluating the usability of the ES, possibly using techniques such as Evaluative Classification of Mismatch [Booth, 1990 & 1991].

- *Saleable System*

  This is the final deliverable version of the ES. The term “saleable” does not necessarily imply that the system will be sold for money. Instead the term is used to mean that the system may have to be “sold” to people who are not committed to its use (*e.g.*, given to other departments or other sites within an organisation). Its release involves the production of user documentation, training materials, and help lines (if required). The ES will have been introduced to a wider community (*e.g.*, alpha and beta releases). Appropriate changes will have been made and system bugs fixed.

- *In Use*

  This ensures that the system is used correctly by checking that the clients, users, and their organisations understand the strengths and weaknesses of the ES. Importantly, it also involves maintaining the knowledge base and updating the functionality of the system on a regular basis. This ensures that the ES remains in beneficial use over the longest possible time, thereby maximising the return on the investment in its development. The deliverables of this stage are the continuing business benefits that the organisation receives from using the system.

The CCA is an “evolutionary” methodology insofar as it supports the continuing evolution of an ES and because it is accepted that the methodology itself will evolve with time. The stages of the CCA state what should be delivered during the ES project. The CCA does not prescribe how each deliverable should be met or how the ES developers should work, and what tools and techniques they should use. The stakeholders in the project are free to use their own experience to decide this. However, the CCA does offer advice and guidance, and it can help the stakeholders plan for each deliverable and manage the project. This is described below.

3. THE SPIRAL MODEL

Boehm's spiral model for project management [1986 & 1989] is gaining in popularity, even among conventional software developers. It is specifically designed to manage risk and was being used by the KADS-II consortium [Bright *et al*., 1991; Killin *et al*., 1991]. Risk may be defined as, “*an event or situation that will have a negative impact on the* *project goals, schedules or budgets and whose probability is not known"* [Bright *et al.*, 1991].

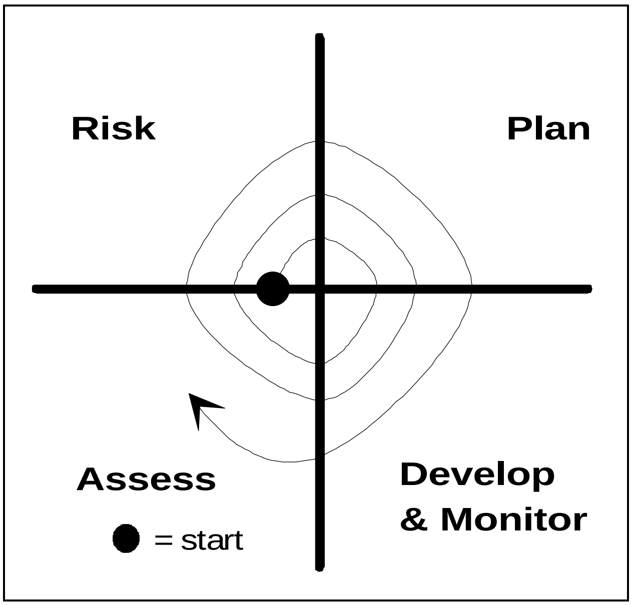

Figure 1. Boehm's Spiral Model

Essentially, the spiral model, shown in Figure 2, provides a way of visualising risk management. The process is divided into quadrants that represent management activities. In the first quadrant (Risk) the project team and their management assess areas of risk that may affect the project by compiling a list of risks. Boehm provides ten areas:

- personnel shortfalls,
- unrealistic schedules and budgets,
- developing the wrong software functions,
- developing the wrong user interface,
- gold plating of functions,
- continuous requirement changes,
- shortfalls in externally provided components,
- shortfalls in performed tasks,
- real time performance capabilities, and
- straining computer science capabilities.

To this list one may add risks that are specific to ES projects or to one's individual circumstances, for example

- poor access to experts,
- uncooperative experts,
- lack of commitment of users, and
- variation in user population.

After compiling a list of risks, managers should rate their assessment of each risk. One can use LOW, MEDIUM, and HIGH, 1 to 5 or any convenient rating. Having prepared this estimate of risk the managers next consider what options can be taken to reduce the risk in high areas.

Initially all possible options should be considered, including completely fanciful ones. Each option is then considered in turn against the project's known constraints. This quickly removes those that are impractical leaving options that could benefit the project.

At this point, one enters the second quadrant (Plan) as the managers prepare a plan for the next cycle. This plan will include the new options decided upon during the Risk quadrant. The plan should outline the targets for the cycle, including deliverables, responsibilities, and budgets if necessary. This results in a new work package plan that each member of the development team can use.

During the third quadrant (Development & Monitor) development work continues and is monitored daily by the project leader if necessary. This quadrant concludes with the production of project progress reports, and deliverable software and documentation as specified in the work package plan.

The progress reports and deliverables are then assessed by project managers during the assessment quadrant to see if they meet expectations. This process can be carried out by a steering or review committee if one has been established to guide the project. This is followed by further cycles of risk assessment, planning, development, and assessment until the system is embedded in use.

Although the spiral model is divided into four equal quadrants, these will not occupy equal lengths of time. The time spent on development will be greater than all the others. However, the spiral model shows that each activity is equally important to the success of the project.

4. THE HELIX METHOD

A large ES shell producer and consultancy has warned that the spiral model can confuse managers, and misrepresents progress, since it implies that the project is going round in circles [*pers comm.* Klahr, 1991]. The spiral model can be represented more intuitively, as shown in Figure 3, as a helix showing that the project does not continually cover the same ground, and that the development is progressing towards the project's goals. It also demonstrates (as the original spiral model does) that the management activities are regularly repeated during the project's development.

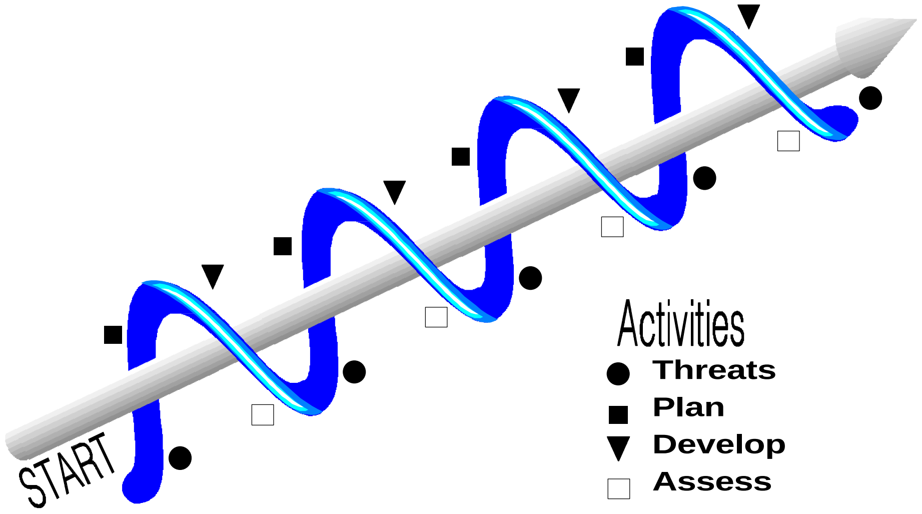

Figure 3. The Helix Method ***(a different view of Boehm's Spiral)***

5. DOCUMENTATION

Assessment of the project's progress is a key activity within the helix or spiral models. To do this successfully, managers need the following documents:

- a clear statement of the objectives of the project,
- a clear statement of the expected benefits,
- clear milestones with planned deliverables, and
- clearly defined roles for all the stakeholders in the project.

Some of these will be represented as project documents. This documentation lets managers make informed decisions based on progress. The documentation also becomes a permanent record of the project and is a vital resource during ED2 when the system is maintained.

KADS is an expert system development method that is gaining in popularity. It describes a very comprehensive set of project documentation [Bright *et al*., 1991]. However, KADS is designed for and by large organisations (*e.g.*, Siemens, Touche Ross, Lloyd's Register) and may not be suitable for use by smaller companies and smaller projects [*pers. comm.* Wielinga, 1991]. Developers using KADS sometimes make statements like,

> *"KADS is exactly the kind of methodology that one can follow most by slavishly*
> [Killin *et al*. 1991].

This implies that developers tend to "pick and choose" elements of KADS that they find useful. The documentation set that accompanies the KADS method involves a minimum of fifty documents (and in practice many more). These range from documents that detail the motivation and aims of the project through to documents that define every functional block within the system.

Whilst we believe that comprehensive project documentation is useful, the heavy documentation burden of KADS is one aspect that reduces its suitability for SMEs. In particular, small development teams would find the generation and maintenance of this documentation very time consuming. We are not advising developers to ignore documentation. Instead we are proposing a rational approach that combines effective (but possibly minimal) documentation with efficient use of project resources.

Section 7 outlines the contents of the documents that should accompany the CCA. These documents need not be lengthy, but each item should be addressed if only to say that it is not applicable. These documents are produced when necessary and may be amended to reflect changes in the evolving system. The management activities that accompany each stage are also described in section 7. Both the content of the documentation and the management activities are derived from those of the KADS-II Framework [Bright *et al.*, 1991] and from Boehm [1986 & 1989]. However, they have been simplified to suit the more limited resources of SMEs.

6. MAINTAINING THE SYSTEM

Because knowledge changes over time, maintaining ESs is significantly different from maintaining conventional systems [Chee & Power, 1990; Pau & Kristinsson, 1990, Killin *et al.*, 1991; Bench-Capon & Coenen, 1992; Coenen & Bench-Capon, 1992]. There may be many maintenance cycles within ED2. Depending on the resources available, the nature of the system, and its expected life, each management cycle may occur every quarter, bi-annually, annually, or at other periods. However, regular management meetings should be scheduled to plan the maintenance of the system. The following documentation should be in place by this stage:

System Documentation. This includes the location of all the project documents: the requirements, architectural, functional and knowledge specifications, along with source code. It should also record what changes have been made to the system’s functionality or knowledge, why they were made, when, and by whom.

User Documentation. This records the latest version of the user documentation that accompanies any system updates, including training materials, installation instructions, trouble shooting advice, and work-arounds.

7. REVIEW OF THE HELIX METHOD

As with other aspects of the CCA we do not intend to be prescriptive (*i.e.*, the CCA states “what” and “when” actions should be taken, not “how” they are performed). Details will vary between projects and with individual management styles. Indeed, on small projects it is feasible for the system to be its own functional specification and for its knowledge base to be the statement of knowledge included in it. Essentially, the project management documentation should be made up of the following components at each stage:

- a document recording the conclusions of the progress assessment,
- a document describing the current requirements and architecture specifications of the developing system (in smaller projects the knowledge base of the system can form this document itself),
- a document describing the knowledge included in the current system (in smaller projects the knowledge base of the system can form this document itself),
- an interim “optimistic” project plan (*i.e.*, what could be achieved in a perfect world),
- a document describing the threats assessment, and
- a “realistic” project plan for the next stage detailing deliverables and task allocation.

These reflect the management activities that occur during each project management cycle. These activities or tasks are described as follows:

- assess progress,
- prepare interim project plan,
- identify threats,
- consider alternatives,
- consider constraints,
- select valid alternatives,
- prepare plan,
- gain acceptance, and
- develop & monitor.

It is not essential for developers following the CCA to use the helix method, but it does provide a way of managing the uncertainty inherent in developing expert systems. The management activities need not be time consuming. For each cycle they may be reduced to just a few hours. The documentation can also be reduced, so that the developing system is its own specification. However, developers should remember that the lack of system documentation may later become a threat to the successful maintenance of the system.

8. CONCLUSION

The helix method of project management recognises that conventional project planning is not always suitable for the inherently uncertain and risky process of implementing an expert system. Having accepted this the helix method allows plans to be made for each development stage within the CCA. At its simplest, there could be one management cycle round the helix for each deliverable within the CCA. The helix method combined with the CCA has the following potential advantages:

- It closely involves all the stakeholders in the system in the development process.
- It provides a clearly defined set of natural milestones for development, which can serve as auditing points.
- It provides a project management technique that guides development stage by stage.
- It visualises the threats to the project's success, reducing the risk of costly failure.
- It is directly suitable for smaller organisations since it does not overburden developers either with exhaustive documentation or time consuming management techniques.
- It accepts that an expert system's knowledge base requires ongoing maintenance and provides a way of managing that maintenance.

Although this project management method has been informed by KADS-II and especially Boehm, it places great emphasis on the reduction of effort, particularly regarding documentation. This is an acceptance that small developers may not have the resources to support a method as exhaustive as KADS but will still benefit from a staged method. Moreover it explicitly recognises that the development of an expert system does not stop once it is brought into use, and that its maintenance will require planning.

9. ACKNOWLEDGEMENTS

This research was funded under the IEATP Program, Project No. IED4/1/2062. We would also like to acknowledge our collaborators: The Royal Institution of Chartered Surveyors, Inference Europe Ltd., Imaginor Systems, and the members of our project user group.

10. REFERENCES

Basden, A. 1989, *A Client Centred Methodology for Building Expert Systems*, in **People and Computers, V**. Sutcliffe, A., & Macaulay, L. (Eds.), Cambridge University Press, Cambridge, UK.

Basden. A., Watson, I.D., & Brandon, P.S. (1991). *The evolutionary development of expert systems.* In: **Research & Development in Expert Systems VIII**, (eds. Graham, I.M., & Milne, R.W.), pp.67-81 Cambridge University Press, Cambridge, UK.

Bench-Capon, T.J.M., & Coenen, F. (1992).
The Maintenance of Legal Knowledge Based Systems. **AI Review**, Vol **6**: pp.129-43.

Boehm, B. (1986).
A spiral model for software development and enhancement. **IEEE Computer**, May 1988.

Boehm, B. (1989).
Software Risk Management. IEEE Computer Society Press.

Booth, P.A. (1990).
ECM: A scheme for analysing user-system errors. In Diaper, D. et al., (Eds.), Human-Computer Interaction - Interact '90: Proc. of the 3rd. IFIP Conf. on HCI, pp.47-54. Elsevier Science Publishers B.V. (North-Holland).

Booth, P.A., (1991).
Errors and theory in human-computer interaction. **Acta Psychologica**, **78**: pp.69-96.

Bright, C., Martil, R., Williams, D., & Rajan, T. (1991).
The KADS-II Framework for KBS Project Management. Proc. Ist SGES Int. Workshop on Knowledge-Based Systems Methodologies.

Chee, C.W.J., & Power, M.A. (1990).
Expert Systems Maintainability. Proc. of the Annual Reliability & Maintainability Symposium, pp.415-18.

Coenen, F., & Bench-Capon, T.J.M. (1992).
Maintenance & Maintainability in Regulation based Systems. ICL Technical Journal, May, pp.76-84.

Colebourne, A., Sawyer, P., & Sommerville, I. (1992).
Evolutionary Development of Interactive Systems. Dept. of Computing, Lancaster University, UK. pers comm.

Diaper, D. (1987).
POMESS: a People Orientated Methodology for expert System Specification, in Proc. 1st. European Workshop on Knowledge Acquisition for Knowledge Based Systems. Addis, T., Boose, J., & Gains, B. (Eds.).

Diaper, D. (1989).
Knowledge Elicitation: Principles, Techniques & Applications. Ellis Horwood Ltd., Chichester, UK.

Department of Trade and Industry (1992).
Knowledge Based Systems Survey of UK Applications. Report performed by Touche Ross and commisioned by the DTI, February, 1992

Inder, R., & Filby, I. (1991).
Survey of Methodologies & Supporting Tools. Proc. Ist SGES Int. Workshop on Knowledge-Based Systems Methodologies.

Killin, J., Morgan-Gray, L., & Porter, D. (1991).
Knowledge Engineering Within Software Engineering - Similarities & Differences. Proc. Ist SGES Int. Workshop on Knowledge-Based Systems Methodologies.

Klahr, P., (1991).
Strategic Implications of KBS Methodologies. Proc. Ist SGES Int. Workshop on Knowledge-Based Systems Methodologies.

Moores, T.T., & Edwards, J.S. (1992).
Could large UK corporations and computing companies use Software Cost Estimating tools? A survey. In, **European Journal of Information Systems**, in print.

Pau, L.F., & Kristinsson, J.B. (1989).
SOFTM: A Software Maintenance Expert System in Prolog. **J. of Software Maintenance: Research & Practice**, **2**(ii): pp.87-111.

Taylor, R.M., Bright, C., Martil, R., & de Hoog, R. (1991).
The management of knowledge-based systems development and maintenance under KADS-II. In: **Research & Development in Expert Systems V111**, (eds. Graham, I.M., & Milne, R.W.), pp.52-66. Cambridge University Press, Cambridge, UK.

Thomas, M. (1991).
What Constitutes a KBS Methodology. Proc. Ist SGES Int. Workshop on Knowledge-Based Systems Methodologies.

Watson, I.D., Basden, A., & Brandon, P. (1992).
A client centred approach to the development of expert systems. In, **Proc. 12th. International. Workshop on Artificial Intelligence, Expert Systems & Natural Language**, in print.