Managing Knowledge in IT Organisations – A Solution for Corporate Support Centres

Debbie Richards and Megan Vazey

Department of Computing

Macquarie University

North Ryde NSW 2109

richards@ics.mq.edu.au, megan@excelan.com.au

Abstract

Knowledge can be difficult to capture and codify, interpret and reuse. Knowledge management is an important determinant of an organisation's competitive edge. It can impact organisational agility and hence customer satisfaction. The rise of the corporate call / support centre and help-desk bears witness to this phenomenon.

In this paper we consider some of the knowledge management issues facing the corporate support centre. We use the support centre of a large multinational high-tech organisation dealing with the complexities of the burgeoning Information Technology (IT) field as our example.

While vendor solutions focus on tracking incoming problem cases, and separately tracking and archiving solutions in a corporate knowledge-base, we observe that there is a huge stone left unturned in the form of problem diagnosis, where-to-search, and what-to-search-for knowledge.

As a solution we offer the ripple down rules (RDR) knowledge acquisition technique, which has addressed many of the shortcomings of first generation expert systems. We review some of the key RDR implementation issues and describe our planned extensions to fit the complexity and volatility of the support centre.

Introduction

The move from product-based to service-based industries in developed countries is clearly seen in the central role now played by an organisation's support centre. In many cases the success of the organisation will depend not on the superiority of their product but on how well they handle support for that product. This is particularly true for the IT industry due to the complex nature of IT products.

We are presently involved in a project to improve the success of a sizeable multi-national support -centre operating in the IT industry. In this context, success can be measured by the effectiveness and efficiency by which customer problems are handled, for example: reduced problem incidence, increased customer self-service, increased automation of problem diagnosis and solution matching, increased accuracy of solution matching as measured by reduced case revisits, increased solution re-use, reduced duplication of solutions, rapid fault and enquiry resolution times, increased customer satisfaction, increased in-line self-learning by call-centre staff, increased staff satisfaction, and reduced staff turnover.

The knowledge management technology used in our support -centre needs to push past the limitations of past-generation knowledge management systems to provide an exhaustive and complete approach to collaborative and knowledge centred support that deals with the complexity and constant change of the environment being troubleshot. It also needs to integrate neatly into the natural workflow of the support centre, so that it saves staff and customers time, and offers them better results.

This paper is organized as follows: in the next section we identify the problems that IT organisations face in their management of software and hardware products; existing technical solutions that assist information management; the trend from information to knowledge management; the progress offered by expert systems in the 1980s; the shortfall of existing vendor solutions; and the key issues to address to achieve the success of our proposed solution in the support centre. We then introduce the Ripple Down Rules technology that we have adopted; and an overview of various implementations based on that approach. Following that we describe the extensions needed to support the specifics of the call-centre domain. In the final section we provide our conclusions and the current state of the project.

The Problem

Knowledge Management for Software and Hardware

The inherent difficulties in software, identified by Brooks (1987) as complexity, conformity, changeability and invisibility, have ramifications not only for software engineering but for the management of knowledge related to that software. Since Brooks' landmark paper, the need to both change and conform to complex environments has increased beyond all expectations. For example, in earlier times, acceptance, integration and stress testing were performed with users, hardware, platforms, applications, inputs and throughput that could be identified before the project started. For many systems that is no longer realistic. Old strategies such as user training to compensate for product shortcomings, designed to pass on the bridging knowledge, are no longer viable in cross-vendor and e-commerce applications.

Knowledge management is not just a problem for software. Managing knowledge about hardware has become more difficult since the (relatively) simple mainframe of the 80's, has been replaced in the 90's with smaller, less expensive open system and windows servers, which can be inexpensively clustered and failed-over as needed, along with dramatic improvements in disk drive capacity. Nowadays the most critical issue is usually data unavailability, data loss, or poor performance, rather than the loss of a single host or server. Discovering the cause of these can be both time consuming and difficult, in complex environments involving multiple vendors, machines, software products and topologies, in an infinite number of combinations. It is no longer possible to expect a single expert to quickly find and resolve such issues. A better approach is needed, to allow both the accumulation of knowledge with guided trouble shooting techniques, along with interfaces to all other relevant knowledge bases.

Technical Solutions to Information Management

Information Technology problems in the past have been primarily addressed via technological solutions. Database technology has provided us with the means to model and capture vast amounts of data. Transaction processing systems and decision support systems have assisted us in turning that data into useful information. Networking technology has allowed data and information to flow freely and enabled the rapid and pervasive uptake of the internet and e-business applications.

From Information to Knowledge

It is widely recognized that the next key step is to go beyond data and information to knowledge management. But, unlike data that can be collected and information that can be summarized and sorted, knowledge is not easily codified and reused. While in the 1980s, and more recently with the huge interest in data mining, there was a view that knowledge could be acquired like nuggets of gold, the contextual, evolving and socially situated nature of knowledge (Clancey 1997) is becoming increasingly apparent. Additionally, the human element in storing, interpreting, defining, and applying the knowledge can not be underestimated. It could be said that "knowledge is in the eye of the beholder".

Expert Systems

The desire to capture and manage knowledge is not a new endeavour. Just as the 1980s was a time of significant activity and advancement in the database and decision support fields, we also saw the launch of the Expert System that promised to reduce reliance on human expertise by capturing knowledge from a domain expert. The Expert System became the repository of accumulated knowledge regarding what error situations have occurred and how they could be resolved. Since the introduction of Expert System technology there have been many successful systems such as XCON (McDermott 1982) and (Eskow 1990).

However, first generation Expert Systems suffered from such problems as giving poor or wrong advice when queried outside their very narrow domain (the brittleness problem) or exponential growth in the time taken to maintain the system (the maintenance problem) (Ignizio 1991).

Knowledge does not remain static, and for most systems it is necessary to change, augment, or update the rule base at regular intervals. In particular the area of knowledge acquisition remains a serious bottleneck in the rapid and effective development of Expert Systems. The knowledge acquisition process is a time-consuming and expensive part of the task of deploying an Expert System in a new domain.

Vendor Solutions

Technological solutions to the Help Desk / Support Centre have been offered by countless software vendors, many of them offering some degree of intelligence. http://www.helpdesk.com (2004) lists 314 vendors of "Help Desk" software, 26 vendors of "Knowledge Management" software, 133 vendors of "CRM and Support Centre" software, and 7 vendors of "Defect Tracking" software. Similarly, http://www.helpdesksoftware.org (2004) has an extensive list of software vendors providing knowledge management software for the helpdesk.

Currently, significant insight is required by call centre staff to either imagine or remember, and then locate existing solutions for a given class of problem:

- Firstly, staff must determine the nature of the problem.

- Secondly, staff must decide where to search, for example on a favourite web page, using a web search engine, or in some other knowledge repository such as a corporate knowledge base.

- Thirdly, staff must determine which search criteria to apply.

It is the capture, refinement and reuse of all three of these types of knowledge: the problem determination knowledge, the where-to-search knowledge, and the what-to-search-for knowledge, that is central to our approach. We believe that the capture of this explicit and, perhaps more importantly, tacit problem-solving knowledge is presently missing from vendor solutions.

A key differentiating feature of our solution is that it offers a closed-loop feedback system whereby users continually and explicitly update and refine system conclusions, and hence the system knowledge as part of their daily work effort. As far as we can tell, existing vendor solutions do not close the loop on system searches, or else provide only very loose closure, and hence build little or no expertise and refinement in the accuracy of query hits.

Our system allows users to record, retrieve, review, refine and rate troubleshooting knowledge in the context of specific problem cases, and using existing solutions. Where-as a search on an existing corporate knowledge base might return multiple solutions irrelevant to the problem on-hand, we aim to capture the minimum set of knowledge required to solve specific classes of customer problems.

The Key Issues to Address for Success

In finding a solution to the call-centre's need to manage knowledge from a vast array of sources we have considered a number of issues, some of which are applicable in other domains. These issues include the need to provide a strategy that supports easy maintenance that can be performed by multiple call-centre staff with domain expertise rather than a single knowledge engineer. The approach needs to be intuitive and easy to learn due to the high turnover of staff within such centres. As well, the system itself needs to provide enough incentive to ensure that staff will use it, benefit from it, enjoy it, and maintain it.

Acceptance of the system will largely depend on successful workflow integration and possibly some workflow redesign where inefficiencies currently exist. Acceptance of knowledge-based systems also hinges on whether the knowledge they contain is deemed to be accessible, valuable and credible.

Incoming problems are often solved via reference to a range of materials, including databases and documents from other vendors, and often involve a chain of people who may contribute to one or more (possible) solutions. Therefore feedback and collaboration become important parts of the process.

Technical issues are often easier to deal with. Thus we introduce first the knowledge representation and acquisition technique, known as ripple down rules (RDR), that we will be using to handle the knowledge. We then consider what strategies are needed to modify the previous implementations to address the softer issues.

In the incomplete and dynamic knowledge environment presented by the support-centre, integration with the way people work, their workflow, and the natural incentives that motivate people to use a system, is essential for system success.

Ripple Down Rules (RDR)

A Brief History

Ripple Down Rules (RDR) were developed in the late 80s (Compton and Jansen 1989) in response to the difficulties associated with acquiring and maintaining a rule-based system. It was observed that pathology experts had difficulty describing what they knew but that they were good at looking at a case and saying how they would handle it. When asked 'why' they used attributes in the case to justify their conclusion. It was further noted that when a similar case was presented, the expert was quickly able to identify whether the same conclusion applied, or if a different conclusion was recommended then they would again pick a feature which distinguished the current case from the case that had prompted the original conclusion.

This process is mimicked by the RDR knowledge acquisition technique. The use of cases to prompt and validate the acquisition of the knowledge and the use of (exception) rules to define and refine the knowledge has led to a knowledge based approach that is both case-based and rule-based and which uses an exception structure for knowledge representation. See (Kang, Compton and Preston 1995) for details of the representation.

The simplicity of the approach allows domain experts to be solely responsible for entering and maintaining the knowledge (Compton et al 1991). A knowledge engineer is only needed in the initial phases to assist with structuring of cases and identification of features. Where cases already exist, as in many domains including pathology, the knowledge engineer is not really needed for set up either.

The approach is designed to allow incremental acquisition and validation of knowledge. In the first deployed pathology system the system went into production with only 100 rules and grew to over 2000 rules in the following four years (Edwards et al. 1993). More recent pathology knowledge bases have been developed solely by pathologists and have grown up to 7000 rules, at a rate of one rule per minute (Lazarus 2000). The knowledge base grows and evolves as more cases are seen.

Variations on the RDR theme

Over the past decade there have been numerous implementations of RDR, and equally numerous variations to the RDR theme. Each one has essentially acquired knowledge in the same manner, by presenting cases to an expert who accepts a conclusion or creates a new rule with the correct conclusion that is attached to the incorrect rule. The most relevant to our implementation are described below.

RRDR and PRDR

Recursive RDR (RRDR) required a modification to the inference process that involved repeated inference cycles using the single classification RDR structure. On each cycle the last true conclusions at the end of each path were returned. A heuristic is then used to choose which conclusions to accept and these are used as inputs and the inference mechanism commences from the beginning again.

Another implementation, which was an initial and only partial solution to the ion chromatography configuration problem, combined RRDR with Possible RDR (PRDR) (Mulholland et al 1993) and reported all last true conditions after taking any branches that might possibly be true. PRDR considers a subset of the branches used in RRDR. Both of these techniques can be difficult to manage as often too many alternatives are provided and it is hard to determine the useful ones.

IRDR

Interactive RDR (IRDR) is a technique that allows the RDR system to prompt the user for more information when required. The user may enter a value or indicate that the value is “unavailable”. The response is added to the case that is then re-evaluated. When changes are made to the case, no matter how small, it is possible that another path will be found through the tree. Some simulation studies have been done using IRDR that yielded poor results. It is conjectured (Paul Compton, personal communication) that it is very difficult for the user to anticipate how the path will change and therefore determining the appropriate change to an A-V pair to test out predictions is very complex.

A more recent development to provide a consultation-style system is the use of rules that are added by the expert to state what A-V pair should be requested next when the value of a certain attribute is known. This is a fairly primitive approach and there are plans to interactively compute the information gain to determine the key attributes that should be queried.

MCRDR

MCRDR was first implemented on the Macintosh platform but the development of MCRDR for Windows for the PC was begun in 1995. Standard tools included in implementations of RDR are the ability to browse and visualise the tree, browse individual traces and examine rules and their status based on the current case. Various statistics are available such as a count of the number of times a conclusion is found in rules, the complexity of individual rules (the number of conditions in the antecedent), the frequency that whole conditions occur, the full complexity of a path, the number of rules in spine sub-trees and a list of conclusions and which rules use them. Numerous other statistics can possibly be calculated.

Shared Features

The various implementations described above indicate the versatility of the RDR structure. The differences tended to concern how the knowledge was being presented and manipulated and the inferencing strategy. Generally, each variation still used:

- cases to provide context, assist in forming rules and for validation;
- the exception structure;
- the use of some form of pre-processor of the raw data;
- incremental KA and maintenance; and
- the development of a simple model in terms of A-V pairs and conclusions.

Our Solution - Extending MCRDR

This project involves an extension of the MCRDR knowledge acquisition technique for the call-centre domain. While some research (Kang et al. 1996, Kim, Compton and Kang 1999 and Kim 2003) has applied MCRDR to the help desk area their work only considers updates by a single expert and our domain is much more complex and dynamic. To name a few of the challenges to existing MCRDR, we need to provide a solution that supports evolution of cases, not just rules, multiple users viewing and maintaining the knowledge, interfaces to numerous other sources and compatibility with current workflow practices.

The modifications needed to existing MCRDR include: Relaxing the Case Differentiation Test for new RuleNodes; the introduction and implementation of the concepts of Live and Registered Conclusions; recording a change history per case and per RuleNode; handling numerous cases via rule tree optimization; handling multiple levels of conclusion granularity; handling conclusion expiration; managing approvals; and building credibility. We do not go into further technical description in this paper. More detail is given in (Vazey and Richards, 2004).

We believe that readers would be most interested in the organisational impact of the system on staff workflow, motivation and acceptance. In the following subsections, we consider issues related to problem determination, conclusion ripple-down, feedback and collaboration, managing incentives, accessibility and usability.

Problem Determination – Applying IRDR and RRDR

Existing commercial defect tracking and knowledge management solutions in the call-centre typically require the user to apply a mental filter for each new case to determine the relevant set of A-V pairs, from a long list of features. We aim to reduce the decision burden for users of the system, and thereby speed up the process of problem determination as well as reduce the risk of information overload to the user. In this endeavour, we intend to apply and extend two variations to the RDR theme: Recursive RRDR (Mulholland et al. 1993) with its repeated inference cycles using the single classification RDR structure; and Interactive RDR (IRDR) which allows the RDR system to prompt the user for more information when required. Hence we propose the development of IR-MCRDR.

Our idea is to use IR-MCRDR in a configuration sense. Our system will assist the user in honing the problem definition by using the current case context to prompt the user for more detail about the specific problem being considered. Only relevant A-V entries will be requested of the user, depending on the current case context.

In addition, as more A-V pairs are gathered to define the case, the case will invoke conclusions that lie deeper down the rule path of the decision tree, and the conclusions displayed will become more specific to the particular problem being observed. Our expectation is that this approach will quickly guide users to the most relevant conclusions for the problem case under consideration.

From General to Specific: Conclusion ripple-down with IR-MC^{2}RDR

Problem determination can be interpreted as a classification problem. At the highest level, when the least is known about the problem case on hand, general solutions exist such as "search the internet", or "read this user guide". As more is known about a problem, more specific solutions become relevant such as "look at this particular website" or "read this paragraph on this page of the user guide". The traditional structure of MCRDR does not handle the transition from general to specific all that well since users must either accept both the general and specific conclusion, and risk being overwhelmed by conclusions; or reject a true but more general conclusion and replace it with a more specific conclusion for the given problem class. In the call centre context, we want to minimise the amount of information that users need to wade through, and only present the most pertinent and relevant solutions, yet we don't want to reject true conclusions.

We have catered for this scenario by inventing multiple conclusion, multiple classification RDR (MC^{2}RDR). This structure acknowledges the situation where both the conclusion from a parent RuleNode and from a child RuleNode may apply. It also clarifies the distinction between a classification, and a conclusion: a case may satisfy one or more classifications, where each classification may offer one or more conclusions.

Feedback and Collaboration

Our system solicits feedback from users to rate the quality of knowledge presented in the context of the given case. In the first instance, if any of the solutions or links presented weren't useful, users are asked to reject the old and create a new web resource (document or web page) and train the expert system to present users with the new information when this particular problem type reappears in the future.

We are exploring ways to capture user opinion about the conclusions presented by including a web discussion board / forum page for each case, and separately for each RuleNode.

As well, we are exploring the idea of allowing users to rate and re-order the conclusions shown for a given type of case in order of the average perceived relevance to that particular category of case.

Managing Incentives

As mentioned earlier, it is vital to the success of our tool that it actually gets used. The system provides a structure for knowledge acquisition and reuse. By itself, it does not provide any knowledge. That part is left to the system's users. Therefore it is important to the success of such a system that it be designed at every step to assist and encourage the user to continue to refine and reuse the knowledge.

In that vein, we intend to keep a record of user click-through activity – how many cases they have worked on, how many conclusions they have created, how many conclusions they have registered (or approved in the case of expert users), the average user rating for conclusions that they have created, and so on. We anticipate that making these statistics publicly available up-front will help to clarify the expectations of users in the system i.e. that users should be active and create good conclusions.

The idea is to publicly support user credibility and with that in mind, users may also be rewarded e.g. with frequent user points, or for instance gold, silver and bronze membership statuses.

Wherever a username is presented, a hyperlink to that user's system details will be provided.

Accessibility

The need for accessibility has encouraged us to design our client-server solution with a web-browser front-end. If the system is extended beyond the local service-desk organisation to assist the global service-desk, separate language interfaces (e.g. Cantonese, Spanish, Japanese) may be required.

Usability

In general, usability is a key ingredient to the success of such a system since the more the system gets used, the quicker the knowledge base will mature and the more compelling the content will become. Usability extends to system responsiveness, ergonomics, ease-of-use and the intuitiveness of the interface.

The ability to search the knowledge, and separately navigate through it would be of high value. It may be that some of the case-based reasoning (nearest neighbour algorithm), formal concept analysis (concept lattice) or data-mining techniques used by others will work well in concert with MCRDR to provide useful knowledge paths that will enrich the user experience, particularly when the conclusions presented are insufficient for them to resolve the problem at hand.

The use of natural language processing for the rule statements would doubtless improve the usability of the system, however at this stage it is beyond the scope of the project. Instead, the simple solution suggested by Kang et. al (1996) to display an explanation for each rule condition for the novice user is relatively easy to implement and may be helpful.

Conclusions

There are numerous commercial help-desk applications to support knowledge management using a range of technologies such as case-based and rule-based reasoning systems, collaborative forum software, and knowledge structuring tools such as FAQ builders. Established and emergent technologies include clustering algorithms, neural networks, and genetic algorithms. A significant number of software vendors are already boasting the use of expert system technology to assist with knowledge management. However, we have found that much of what is offered as call centre / troubleshooting / case tracking systems are just sophisticated databases that are able to keep the current status of the case up-to-date and the customer informed. They don't help with problem determination or solution matching.

In our present project we need to work within the existing environment and culture to develop a solution that works with existing case/databases and current work practices. The fact that knowledge in the form of problem cases and solutions is constantly evolving, requires a technique designed for maintenance that can be easily performed by the call centre representative in a manner that fits with the nature of their job. We believe that RDR offers these features and with our proposed extensions can address the requirements of our project. Currently we have a prototype that we are in the process of connecting to the organisation's case-bases.

Getting update-access and even read-access to these corporate-wide (overseas-controlled) resources has not been easy and remains our current greatest obstacle between realizing a solution that could or does make a difference. As is often the case, knowledge management solutions require changes at the strategic level before changes at the operations level have a chance.

References

Brooks, F. P. (1987) No Silver Bullet: Essence and Accidents of Software Engineering, IEEE Software, 4:10-19.

Clancey, W. (1997) The Conceptual Nature of Knowledge, Situations and Activity In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds) Expertise in Context: Human and Machine AAAI Press/MIT Press, Cambridge, Massachusetts, 247-291.

Compton, P. J. and R. Jansen (1989). A philosophical basis for knowledge acquisition. 3rd European Knowledge Acquisition for Knowledge-Based Systems Workshop, Paris: 75-89.

Compton, P., Srinivasan, A., Edwards, G., Malor, R. and Lazarus, L., (1991) Knowledge Base Maintenance Without a Knowledge Engineer Proceedings World Congress on Expert Systems, Pergmon Press, Orlando, 1:668-675.

Doyle, D., Tsymbal, A. and Cunningham, P. (2003). A Review of Explanation and Explanation in Case-Based Reasoning. D2003, TCD CS report. http://www.cs.tcd.ie/publications/tech-reports/reports.03/TCD-CS-2003-41.pdf

Edwards, Glenn (1996) Reflective Expert Systems in Clinical Pathology MD Thesis, University of New South Wales.

Edwards, G. Compton, P., Malor, R., Srinivasan, A. and Lazarus, L. (1993) PEIRS: a Pathologist Maintained Expert System for Interpretation of Chemical Pathology Reports Pathology 25:27-34.

Edwards, G., Kang, B., Preston, P. and Compton, P. (1995) Prudent Expert Systems with Credentials: Managing the expertise of Decision Support Systems Int. Journal Biomedical Computing 40:125-132.

Eskow, D. (1990) Wendy's Turns to PCs to Beef Up Operations PC Week, February 12, 1990, 121-124.

Ignizio, James, P. (1991) Introduction to Expert Systems: The Development and Implementation of Rule-Based Expert Systems McGraw-Hill Inc, New York.

Kang, B. (1996) Validating Knowledge Acquisition: Multiple Classification Ripple Down Rules PhD Thesis, School of Computer Science and Engineering, University of NSW, Australia.

Kang, B., P. Compton and P. Preston (1995). Multiple Classification Ripple Down Rules : Evaluation and Possibilities. Proceedings of the 9th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, University of Calgary.

Kang, B., Yoshida, K., Motoda, H., Compton, P. and Iwayama, M., (1996) A help desk system with intelligent interface in P Compton, R Mizoguchi, H Motoda & T Menzies, PKAW'96: Pacific Rim Knowledge Acquisition Workshop, 1996 (Sydney, Dept. of Artificial Intelligence, School of Computer Science and Engineering, UNSW, 1996), pp 313-332.

Kim M. (2003) PhD thesis, Document Management and Retrieval for Specialised Domains: An Evolutionary User-Based Approach. UNSW, 2003.

Kim, M., Compton, P., Kang, B. (1999) Incremental development of a web-based help desk system. Proceedings of the 4th Australian Knowledge Acquisition Workshop (AKAW 99), University of NSW, Sydney, 5-6 Dec 1999, 13-29.

Lazarus, L. (2000) Clinical Decision Support Systems: Background and Role in Clinical Support, http://www.pks.com.au/CDSS_White_Paper_doc.pdf

Martinez-Bejar, R., Benjamins, R. and Martin-Rubio, F. (1997) Designing Operators for Constructing Domain Knowledge Ontologies, Enric Plaza and Richard Benjamins (Eds), Knowledge Acquisition, Modeling and Management. Lecture Notes in Artificial Intelligence 1319, Springer-Verlag, 159-173.

Martinez-Bejar, R., Benjamins. R., Compton,, P., Preston, P. and Martin-Rubio, F. (1998) A Formal Framework to Build Domain Knowledge Ontologies for Ripple-Down Rules-Based Systems. In Gaines, B. and Musen, M. (eds) Eleventh Workshop on Knowledge Acquisition, Modeling and Management (KAW'98), Banff, Alberta, Canada, 18-23 April, 1998, Vol 2:SHARE.13.

Martinez-Bejar R., Shiraz G. M., and Compton P. (1998) Using Ripple Down Rules-based Systems for Acquiring Fuzzy Domain Knowledge, In Gaines, B. and Musen, M. (eds) Eleventh Workshop on Knowledge Acquisition, Modeling and Management (KAW'98), Banff, Alberta, Canada, 18-23 April, 1998, Vol 1:KAT.2.

McDermott, J. (1982) R1: A Rule-Based Configurer of Computer Systems Artificial Intelligence (19):38-88.

Mulholland, M., Preston, P., Sammut, C., Hilbert, B. and Compton, P. (1993) An Expert System for Ion Chromatography Developed using Machine Learning and Knowledge in Context Proc. of the 6th Int. Conf. On Industrial and Engineering Applications of Artificial Intelligence and Expert Systems Edinburgh.

Shiraz, G.M. and Sammut, C. (1998) Acquiring Control Knowledge from Examples Using Ripple-down ules and Machine learning, 11th Workshop on Knowledge Acquisition, Modeling and Management, Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:KAT-5.

Vazey, M. and Richards, D. Achieving Rapid Knowledge Acquisition in a High-Volume Call Centre, Proceedings of the Pacific Rim Knowledge Acquistion Workshop, in conjunction with PRICAI’04, Auckland, 9-10th August, 2004.