Evaluation of the FastFIX Prototype 5Cs CARD System

Megan Vazey, Debbie Richards

Department of Computing, Division of Information and Communication Sciences, Macquarie University.
meganv@excelan.com.au, richards@ics.mq.edu.au

Abstract. The 5Cs architecture offers a hybrid Card And Rule-Driven (CARD) system that supports the Collaborative generation and refinement of a relational structure of Cases, ConditionNodes, Classifications, and Conclusions (hence 5Cs). It stretches the Multiple Classification Ripple Down Rules (MCRDR) algorithm and data structure to encompass collaborative classification, classification merging, and classification re-use. As well, it offers a very lightweight collaborative indexing tool that can act as an information broker to knowledge resources across an organisation's Intranet or across the broader Internet, and it supports the coexistence of multiple truths in the knowledge base. This paper reports the results of the software trial of the FastFIX prototype - an early implementation of the 5Cs model, in a 24x7 high-volume ICT support centre.

Keywords: Single Classification Ripple Down Rules, Multiple Classification Ripple Down Rules, SCRDR, MCRDR, Knowledge Engineering, Knowledge Acquisition, CARD, top-down rule-driven, bottom-up case-driven.

Introduction

The 5Cs model is comprised of the Collaborative generation and refinement of a relational structure of Cases, ConditionNodes, Classifications, and Conclusions (hence 5Cs). The 5Cs model is a Case And Rule Driven (CARD) system for Knowledge Acquisition (KA) that uses a Case Oriented Rule Acquisition Language (CORAL) to acquire rules in a similar way to Multiple Classification Ripple Down Rules (MCRDR) (Kang, 1995) and its predecessor the Single Classification Ripple Down Rules (SCRDR) (Compton and Jansen 1989), but with significant extensions. For example, new data structures and algorithms are presented to allow experts to more effectively collaborate in building up both the knowledge and case bases.

The extensions have been motivated by our work in developing a trouble shooting system for a high volume ICT support centre. Knowledge in this domain changes rapidly and is driven by the need to maintain and link problem and solution cases. For this reason we chose to use the MCRDR combined case and rule based approach to incremental KA. However, in this domain we found that the knowledge needed to identify and solve the problem came from many, varied and globally distributed sources, and that the cases themselves were often in a state of flux and needed to be worked up as new information came to light. Sources of information could include: clients, peers, third party vendors or software/hardware engineers and was obtained through personal conversations and notes, Internet sites, manuals and specifications.

The 5Cs system allows knowledge workers to collaboratively refine and expand a topic using an expert systems approach by consistently asking users to confirm, add to, or refine the knowledge presented, typically within the context of a current case. 5Cs supports intuitive KA as it allows the capture and sharing of the questions that experts ask themselves when classifying incoming cases.

This work is similarly motivated to the early work to reconcile multiple sources of expertise that have been captured into multiple MCRDR KBS (Richards and Menzies 1998) using Formal Concept Analysis (Wille 1992). In that work, however, identification and resolution of conflict was not fully integrated into the KA cycle, but performed as desired when another expert view was to be compared and required making changes to the individual sources and regeneration of the combined model for further verification and validation.

More recently Beydoun et al. (2005) have extended Nested RDR (NRDR) (Beydoun and Hoffman 2000) to support cooperative KA. NRDR differs from MCRDR in that it allows multiple SCRDR trees to be combined into an hierarchical conceptual structure. The approach seeks to detect *internal inconsistencies*, which are inconsistencies between multiple experts, and are resolved by the domain expert as independent KBS are individually integrated into the system. To assist, various estimates are derived to determine the probability of internal inconsistency and then determine a "trend of external inconsistencies". The degree of external inconsistency is seen to reflect how well the model represents the world in terms of ontological consistency, completeness and accuracy. As with the work described in this paper, "the intuition is that a coherent collective expertise is a better reflection of 'reality'" (Beydoun et al. 2005, p. 48). Our work differs from this research in that combining expertise is a normal part of each KA cycle to develop a shared and single model, which can contain managed inconsistencies, rather than a technique employed to deal with inconsistencies whenever a new KBS is to be integrated. This is necessary in the 24x7 follow-the-sun call centre environment as multiple individuals, often distributed by time and place, may be involved with specifying the same problem/solution and need a way of working with the knowledge entered by themselves and others over time.

A subset of the features proposed by the 5Cs model has been tested via a prototype system known as FastFIX. FastFIX was developed to support the troubleshooting process in a high-volume and complex 24x7 support center in the ICT domain (Vazey and Richards, 2006). The 5Cs system architecture is presented in the next section. The results of the FastFIX software trial are then evaluated and presented.

2. The 5Cs System

In the 5Cs system, a condition mesh structure is provided, comprising of RuleNodes i.e. (ConditionNodes) as shown in Fig 1. In this structure, each parent RuleNode may have multiple child RuleNodes, and each child RuleNode may have multiple parents. As well, there is an N-to-N relationship between Cases and their live and/or registered ConditionNodes (i.e. RuleNodes); an N-to-N relationship between the ConditionNodes and the Classifications that they represent; and an N-to-N relationship between the resultant Classifications and their Conclusions.

A Case can evaluate to TRUE for multiple Condition paths in the condition mesh, hence a Case can fetch multiple Classifications, where each Classification may be linked to multiple Conclusions. As well, Conclusions can be reused across multiple Classifications, and those Classifications may be reused across Condition paths, and across multiple Cases. Note that in Fig **1** only a subset of classifications and conclusions are shown for RuleNodes 6, 7, and 8 and for simplicity links to the classifications and conclusions at other RuleNodes have not been included in the figure. Note also that in the 5Cs system, the Attributes, Cases, Conditions, Classifications, and Conclusions may each be the subject of Collaborative creation, editing, or deletion. In addition, RuleNodes can be collaboratively relocated.

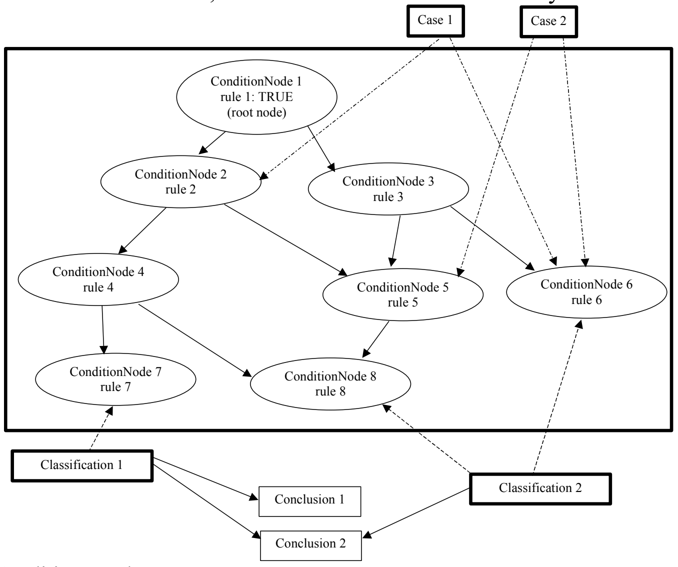

Fig 1: 5Cs Condition Mesh

2.1 Tracking Case-RuleNode Associations

Unlike many MCRDR systems which only require cornerstone cases to be kept, in a domain like the call center where the cases are volatile we keep all cases and track all Case-RuleNode associations. This is somewhat similar to the use of an execution history, tracking of rule usage and the proposed review of rule usage against the case history in the HeurEAKA RDR system (Bekmann and Hoffmann 2004), which uses NRDR and genetic algorithms for channel routing in VLSI design. In the 5Cs model, a “Tracked” Case is one whose Live and Registered RuleNodes are being remembered by the system. A Live RuleNode for a given Case is one that is currently the last `TRUE` RuleNode on a given path through the knowledge base for that case. Its conclusions are part of the set of current conclusions derived from the knowledge base for the case. The system remembers its Live RuleNodes for "Tracked" cases. Live RuleNodes may be correct or incorrect. Correcting incorrect live RuleNodes is the primary role of a (human or computer) expert who trains the knowledge base.

A Registered RuleNode is one that has been confirmed by a User as being correct and `TRUE` for that Case. For each case, each RuleNode registration may be current or expired. The test for RuleNode-Case registration expiry is that if the last modification or creation date on the RuleNode-Case association is more recent than or the same as the last modified dates on the RuleNode and the Case, then the registration is current. Otherwise, the registration is expired. The expiry of registered case-RuleNode relationships is something that can be notified and displayed to users in the summary of cases or RuleNodes of interest to them. The user can also be notified whenever the list of live RuleNodes differs from the list of registered RuleNodes for a given case, or the list of live cases differs from the list of registered cases for a given RuleNode.

In the Pathology Expert Interpretative Reporting System (PEIRS), typographical or conceptual errors in RuleNode expressions were corrected with the use of "fall-through" rules (Edwards, 1996, p. 119), but potentially resulting in corruption of the KBS (Edwards 1996, p.88). Similarly, Kang identifies the situation where the domain knowledge represented by an existing rule tree needs to be changed in such a fashion that a cornerstone case for an existing RuleNode will *drop-down* to a new child RuleNode (Kang, 1995, p.50). He suggests that if absolutely necessary, the rules suggested by the MCRDR difference list for the new RuleNode can be overridden (Kang, 1995, p. 65). The approach was not fully explored (Kang, 1995, p67). Note that cases don't only *drop-down*. They may drop-across from a sibling node; or they can *recoil* to an ancestor RuleNode for example when the editing or relocation of a dependent RuleNode is restrictive enough that the case under review is now excluded.

Unlike in pathology where each report handled a new case and changes to a patients data resulted in a new case with new conclusion, in the Call Centre domain it is likely that information about a customer problem will develop over days or even months in situations where a problem is intermittent or temporarily fixed but reemerges at a later date. A supposed solution to a problem may turn out to not have really fixed the problem at all. This means that the knowledge (ie the rules) and the problem/solution case/s may also need changing. The 5Cs structure and algorithms allow the knowledge base to evolve, including changes to cases, RuleNodes, intermediate conclusions, ontological entry or attributes. This is achieved through tracking of live vs registered case-RuleNode associations.

In fact, FastFIX tracks all changes to the system and identifies users when an inconsistency in the system occurs. The strategy assists with more rapid knowledge acquisition since it can highlight inconsistencies between expert opinions, just as MCRDR supported quicker KA by allowing more than one rule to be acquired for each case. It lets users capitalize on the knowledge acquisition opportunity presented by the case drop-down scenario, and this in turn may result in quicker coverage of the domain and greater learning opportunities for users. The separation of live and registered case-RuleNode associations is a key part of being able to resolve classification conflicts between multiple experts, and even between what a single expert thinks today, as compared with tomorrow (Gaines, 1993).

2.2 The FastFIX Prototype

As mentioned earlier, a subset of the features proposed by the 5Cs model was tested in the ICT support center problem domain via a prototype system known as FastFIX. Significant novel ideas implemented in the FastFIX prototype and tested during the FastFIX software trial included:

- The ability for multiple users to build an MCRDR-based decision tree in a wiki[1]-style collaborative effort. This includes the identification of classes of incoming problem cases and manual indexing of solutions by multiple users using rule conditions equivalent to logical tags in a folksomony[2].
- Reference to multiple exemplar cornerstone cases for each RuleNode.
- The ability for users to edit previously created cases (including cornerstone cases) and RuleNodes in the system.
- Continuous background monitoring of changes to the knowledge base so that users with affected RuleNodes and Cases can notice and respond to the changes. This approach allows classification conflicts to be identified, clarified and resolved and hence it enhances knowledge acquisition.
- The ability for users to "work-up" a case using a novel Interactive and Recursive MCRDR decision structure.
- Separation between classifications and conclusions so that richer classification relationships can be maintained.
- The ability for users to relocate i.e. move RuleNodes in the system.

3. Results of the FastFIX Software Trial

Table 1 summarises user activity during the software trial. 12 users registered themselves, including the author (User ID 12). Most of the registered users were onlookers – managers, team leaders and other interested parties. In addition to the author, there were three main contributors with user IDs 1, 3 and 6. These contributors were able to use the system with minimal training and supervision (less than 60 minutes per contributor). Contributors commenced by providing troubleshooting knowledge for the most frequently occurring sub-domain of troubleshooting errors.

In total 172 cases and 107 RuleNodes were created. The total number of case edits was 139 and the total number of RuleNode edits was 141 demonstrating both the desire and capacity of users to contribute to knowledge evolution in this way. In total there were 104 case drop-throughs resulting from RuleNode creations. As well, there were 96 case drop-through events resulting from RuleNode edits where each of these events may have affected 1 or more cases. **Fig 2** provides a graphical representation of the data in Table 1.

[1] Wikipedia (http://www.wikipedia.org/) defines a Wiki as the collaborative software and resultant web forum that allows users to add content to a website and in addition, to collaboratively edit it. Wikipedia demonstrates the power of the Wiki paradigm.

[2] The term "folksomony"[2] was first coined by Thomas Vander Wal[2] (2005) to describe forums in which people can tag anything that is Internet addressable using their own vocabulary so that it is easy for them to re-find the item.

Table 1: User Activity in the prototype FastFIX system

User ID	1	2	3	4	5	6	7	8	9	10	11	12	Totals
Total Case Creations	64	0	15	2	2	83	1	1	0	0	1	3	**172**
Total RuleNode Creations	42	0	13	0	0	30	1	0	0	0	0	21	**107**
Total Case Edits	45	0	22	0	0	33	0	0	2	0	7	30	**139**
Total RuleNode Edits	32	0	13	0	0	59	0	0	0	0	0	37	**141**
Total Case drop-throughs resulting from RuleNode Creations	59	0	1	0	0	43	0	0	0	0	0	1	**104**
Total Case drop-through Events resulting from RuleNodeEdits	24	0	6	0	0	48	0	0	0	0	0	18	**96**

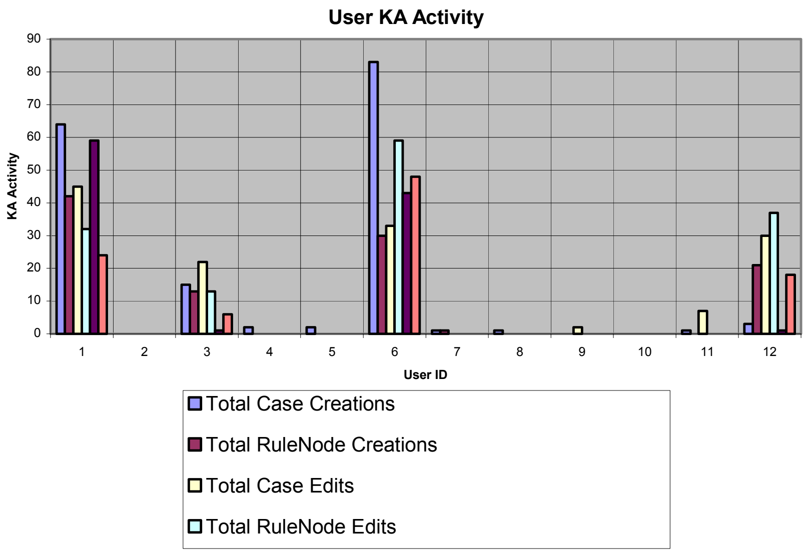

Fig 2: User Activity in the prototype FastFIX system

3.1 Solution Effectiveness

After 7 hours of effort the test team had captured 105 cases and 55 RuleNodes. The red arrow in each of following figures has been used to indicate this point in time. At this point the team had provided enough RuleNodes to automatically solve approximately 90% of errors on errant equipment in the selected error sub-domain. These errors contribute to 30% of all errors seen by the system which account for 20% of the ~5,000 problem cases per day seen by the global ICT support centre. Hence after 7 hours of effort enough knowledge had been acquired to automatically provide solutions to more than 270 cases per day, without requiring the trouble-shooters to figure out the class of problem on hand, where to search for a solution, or what to search for, for example in the corporate solution tracking system.

Say that each case takes on average 15 minutes to solve, and that 1 minute of this time is spent in determining the problem and finding its solution. This represents a time saving of 1 mins * 270 cases per day, or 4.5 hours per day. Actually, the average solution search time is possibly a lot longer. One of the problems with manually searching for solutions is that if you haven't found the answer, you don't know if its just because your not searching for it correctly, or if its because a solution does not exist. The FastFIX system has the advantage that it unambiguously associates relevant solutions with their incoming problem classes. If the answer is unknown, FastFIX can provide that information.

After the first 105 cases and 55 RuleNodes, the test team broadened the knowledge domain being covered to include a new error sub-domain. This evaluation strategy parallels the strategy used in the 4 year PEIRS SCRDR software trial in which additional domains were incorporated incrementally after the pathologists had gained confidence in the performance of the system with the initially selected thyroid domain (Edwards, 1996, p.90). The trial of the prototype ceased after 107 RuleNodes and 172 cases had been accumulated. At that point, no new information was being gathered and attention was turned to additional features to enhance the system.

3.2 Case Drop-throughs

Fig 3 shows the cumulative case and RuleNode creations and Case drop-throughs resulting from RuleNode Creations in greater detail. A unique KA Event ID has been assigned to each unique timestamp captured in this subset of data and it has been used to construct the x-axis. It can be observed in Fig 3 that the first 20 RuleNodes were provided to the system in a top-down (and hence rule-driven linear) manner. In contrast, RuleNodes 21 to 55 were provided mostly on the basis of cases seen in a case-driven bottom-up monotonically increasing and stochastic manner as described in (Vazey 2006). After this point, users were selective in choosing which cases to train the system with, choosing cases that were expected to be novel. Hence RuleNodes 56 to 107 were provided to the system in a more top-down (and hence rule-driven linear) manner as for the first 20 RuleNodes.

It is difficult to say how the ability of users to edit RuleNodes affects the overall case and RuleNode creation trajectories. If most of the RuleNode edits were cosmetic e.g. as a result fixing spelling mistakes then it can be expected that these KA trajectories would be little affected by the RuleNode edits. However, if RuleNode editing represents a significant KA activity whereby genuinely new knowledge is being acquired, rather than existing knowledge being cosmetically corrected, then those RuleNode edit events should be added into the above case-driven KA trajectory. However, it was beyond the scope of this trial to examine this in any detail.

Fig 4 shows the Cumulative Case Creation and RuleNode Edit Curves. The number of RuleNode edits appears to grow in proportion to the number of cases seen by the system, which indicates that RuleNode editing tends to be a top-down knowledge acquisition activity.

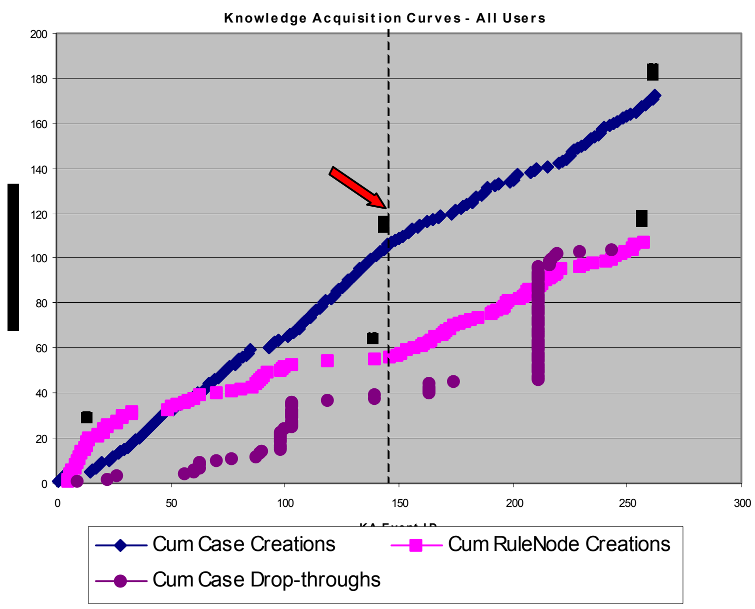

Fig 3: Cumulative Case and RuleNode Creation Curves

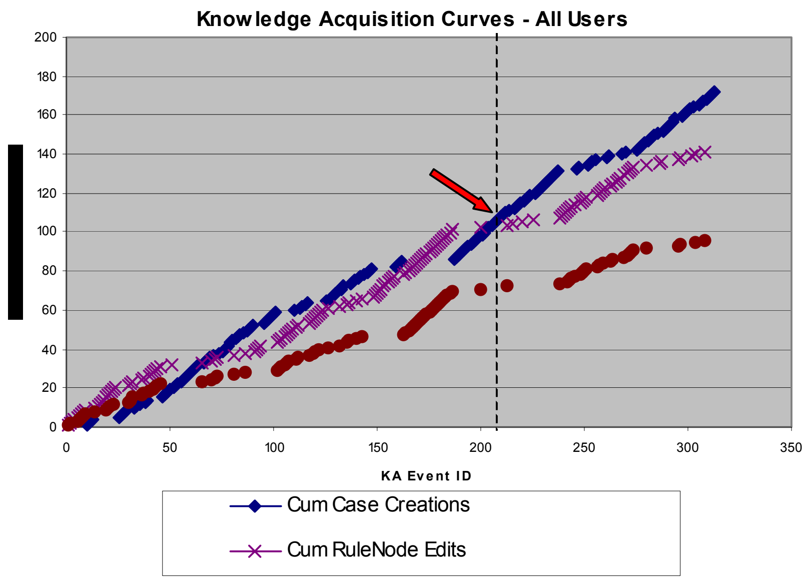

Fig 4. Cumulative Case Creation and RuleNode Edit Curves

3.3 Individual KA Curves

Individual KA Curves are displayed in the next four figures for the 3 most active users (with User IDs 1, 6, and 12) in the system, including the author (User ID 12).

Vertical lines have been included in the graphs to show the co-occurrence of RuleNode Creations and their resultant case drop-downs, as well as RuleNode Edits and their resultant case drop-down events.

Case edit events have been left out of the curves to allow the rate of RuleNode accumulation to be compared with the rate of Case accumulation.

In **Fig 5**, User 1's KA curves show a steady rate of accumulation of both cases and RuleNodes. It appears that RuleNodes are acquired bottom-up prior to the domain change, and top-down thereafter.

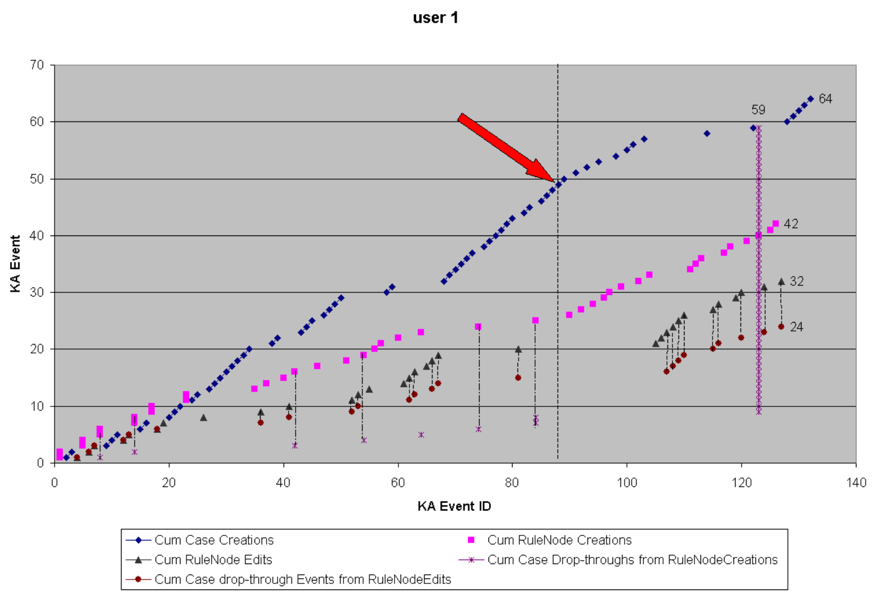

Fig 5: User 1's KA Curves

In **Fig 6**, User 6's KA curves show a steady rate of accumulation of both cases and RuleNodes. As for User 1 it appears that RuleNodes are acquired bottom-up prior to the domain change, and top-down thereafter. We can also see that User 6 undertook a major RuleNode editing activity between KA event 80 and 100. This effort was focussed on widening the scope of the rule statements in a number of RuleNodes.

In **Fig 7**, User 12's (i.e. the author and researcher's) KA curves show a focus on RuleNode edits in the early phases. At this point the system was still under development so both the users and the system were changing in the way they interacted with each-other. User 12 created the first 20 RuleNodes in the system in a top-down fashion after consulting with User 6. In contrast, User 12 was involved in very few case creations.

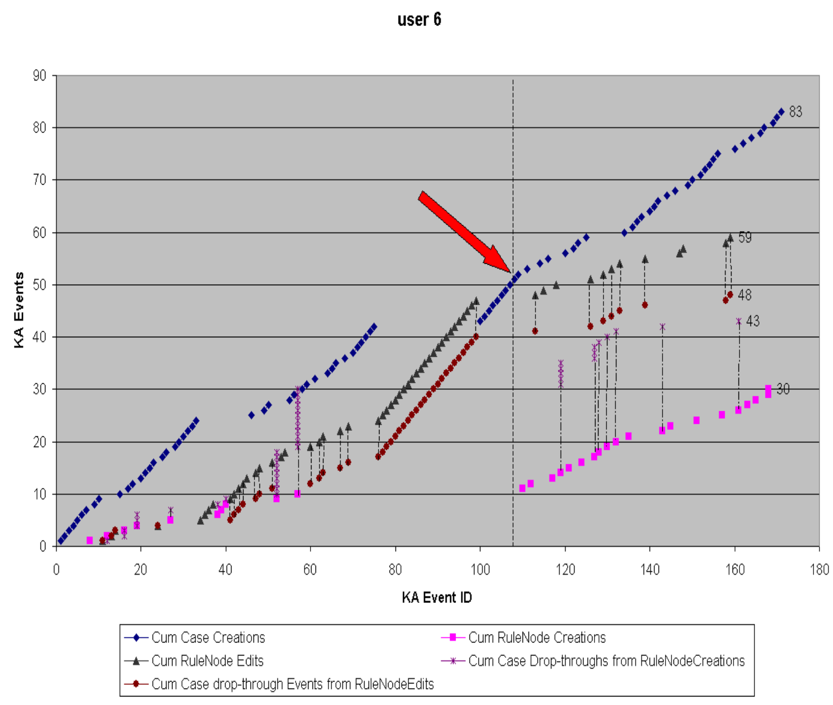

Fig 6: User 6's KA Curves

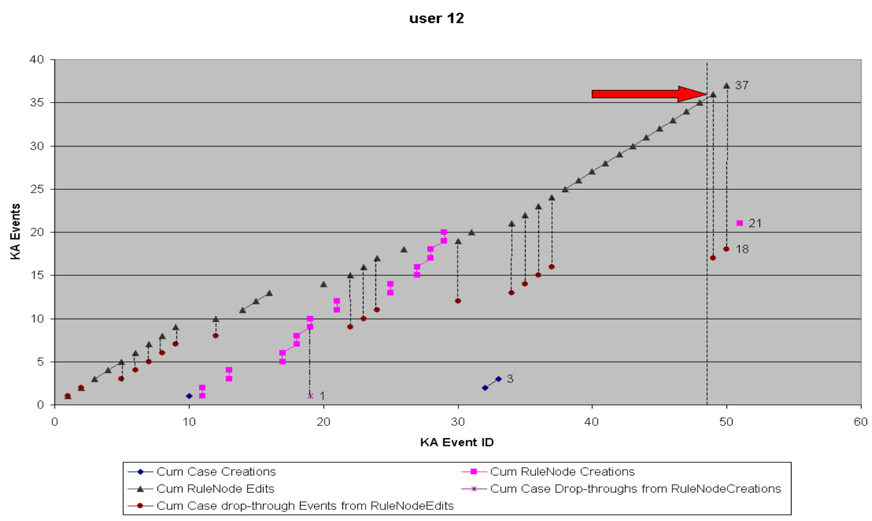

Fig 7: User 12's KA curves (i.e. the author's)

Note that the initial knowledge base activity by user 12 parallels that reported in the early days of the PEIRS trial. In PEIRS the first 198 rules were added off-line while interfacing problems were sorted out (Edwards 1996, Kang 1995). In FastFIX, the first 21 RuleNodes were added in this manner.

Conclusions

In summary, the results of the FastFIX software trial indicate that:

- Users were able to use the system with minimal training and supervision (less than 60 minutes on average per contributor).
- The system was able to support multiple users in collaboratively building the knowledge base, and was effective at communicating to colleagues when changes to the knowledge base occurred that might affect areas of the knowledge base that they had been working on.
- Case drop-downs occurred frequently (in this example, about as frequently as RuleNodes were created and edited), so it appeared to be an important enhancement to separately track the live vs registered RuleNodes for users.
- Given that case drop-downs occurred so frequently, it seemed to be an important enhancement to enable more representative cornerstone cases to be substituted for less representative ones when case drop-down occurred.
- Users were willing and able to work within both a case-based bottom-up and rule-based top-down mindset to edit RuleNodes, and to add knowledge to the decision tree.
- Users took advantage of the ability to label the classification represented by RuleNodes as distinct from the conclusions at that RuleNode. 22 of 106 manually constructed RuleNodes had their classifications labelled.
- Users found cause to refer to conclusions at other RuleNodes even though this feature was only made available at a late stage in the software trial.
- The ability to combine text and hyperlinks in the conclusions at RuleNodes meant that multiple conclusions could be referred to at a single RuleNode.
- Users found cause both to disable some RuleNodes, and to negate the effect of parent RuleNodes under certain rule conditions. Users disabled 10 RuleNodes by editing them and creating rule statements that were `FALSE`. In addition, users created 3 stopping RuleNodes to negate the validity of the parent RuleNode under certain rule conditions.
- Users were able to add new attributes to the system and the system provided a way for users to effectively "work-up" their problem cases via getAttribute() functional conclusions.
- Users were able to add new attributes to the system.
- The system was effective at rapidly acquiring sufficient knowledge to improve the effectiveness and efficiency of troubleshooting in the selected subdomains.

Acknowledgments. This work has been funded via an Australian Research Council Linkage Grant LP0347995 and a Macquarie University RAACE scholarship. Thanks also to my industry sponsor for ongoing financial support.

References

1. Bekmann, J. and Hoffmann, A. (2004) HeurEAKA– A new approach for Adapting GAs to the problem domain, Eds. Zhang, Guesgen, Yeap *PRICAI 2004: Trends in Artificial Intelligence*, Springer, Berlin, 2004, pp. 361 - 372 .
2. Beydoun, G. and Hoffmann, A. (2000). Incremental acquisition of search knowledge. *Journal of Human-Computer Studies*, 52:493.530, 2000.
3. Beydoun, G., Hoffmann, Fernandez Breis, J. T, Martinez Bejar, R. Valencia-Garcia, R. and Aurum, A. (2005) Cooperative Modeling Evaluated, *International Journal of Cooperative Information Systems*, World Scientific, 2005, 14 (1), 45-71.
4. Compton, P., (2000). Simulating Expertise in Compton, P. Hoffman, A. Motoda H. Yamaguchi T, Proceedings of the 6th Pacific Knowledge Acquisition Workshop, Sydney, p51-70.
5. 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.
6. Edwards, G. (1996) Reflective Expert Systems in Clinical Pathology MD Thesis, School of Pathology, University of New South Wales.
7. Gaines B. R., Shaw M. L. G., (1989) Comparing Conceptual Structures: Consensus, Conflict, Correspondence and Contrast.
8. Kang, B. (1996) Validating Knowledge Acquisition: Multiple Classification Ripple Down Rules PhD Thesis, School of Computer Science and Engineering, University of NSW, Australia.
9. 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.
10. Richards, D and Menzies, T. (1998) Extending the SISYPHUS III Experiment from a Knowledge Engineering to a Requirements Engineering Task, Richards D., Menzies, T., Task 11th Workshop on Knowledge Acquisition, Modeling and Management, SRDG Publications, Banff, Canada, 18th-23rd April, 1998.
11. Vazey, M. (2006) Stochastic Foundations for the Case-driven Acquisition of Classification Rules. *EKAW 2006* (in publication).
12. Vazey, M. and Richards, D. (2004) Achieving Rapid Knowledge Acquisition. Proceedings of the Pacific Knowledge Acquisition Workshop (PKAW 2004), in conjunction with The Eighth Pacific Rim International Conference on Artificial Intelligence, August 9-13, 2004, Auckland, New Zealand, 74-86.
13. Vazey, M. and Richards, D. (2006) A Case-Classification-Conclusion 3Cs Approach to Knowledge Acquisition - *Applying a Classification Logic Wiki to the Problem Solving Process*. International Journal of Knowledge Management (IJKM), Vol. 2, Issue 1, pp 72-88; article #ITJ3096, Jan-Mar 2006.
14. Wille, R. (1992) Concept Lattices and Conceptual Knowledge Systems *Computers Math. Applic.* (23) 6-9: 493-515.