Research Issues in Active Database Systems: Report from the Closing Panel at RIDE-ADS '94

Jennifer Widom

Department of Computer Science
Stanford University
Stanford, CA 94305-2140
widom@cs.stanford.edu

1 Introduction

The Fourth International Workshop on Research Issues in Data Engineering was held Feb. 14–15, 1994 in Houston, Texas, in conjunction with the IEEE CS International Conference on Data Engineering. Each year, the RIDE workshop covers a specific research topic within the general area of data engineering; RIDE-ADS '94 focused on Active Database Systems. Over the past several years there has been a tremendous surge of interest in this area, with a rapidly increasing number of research results, prototypes, and support in commercial products. RIDE-ADS '94 was the first workshop to bring together leading researchers, developers, and users in the important area of active database systems.

The workshop included two panels: an opening panel in which current or potential users of active database technology described their application requirements, and a closing panel in which the panelists and ultimately many of the workshop participants attempted to assess whether active database researchers are exploring the right issues from the application's perspective. This short paper summarizes the closing panel of RIDE-ADS '94. Note that this paper does not provide an introduction to or overview of the field of active database systems; for background material see, e.g., [1, 2, 3].

2 Summary

The discussions during the panel stayed largely but not entirely focused on the question of active database research issues from the application perspective. There were nine panelists. Each panelist was asked to prepare brief answers to a set of questions. The sets of answers were discussed by all participants, and finally a number of more general issues were discussed.

The questions asked of the panelists were:

1. Name an application that will certainly be supported by active database systems in the not-too-distant future.
2. Name an application that will certainly not be supported by active database systems in the near future.
3. Name an area of active database systems in which you are not working but that is crucial to meet the needs of applications.
4. Name an area of active database systems that is not on the critical path to supporting applications.
5. Name an area of active database systems that should have been discussed in the course of the workshop but was not.

The nine panelists included three of the active database "users" from the opening panel and six "experts in the active database area." The panelists were:

Renato Barrera, Intergraph
Alex Buchmann, Darmstadt Technical Univ.
Sharma Chakravarthy, Univ. of Florida
Lois Delcambre, Oregon Graduate Institute
Klaus Dittrich, Univ. of Zurich
Rodolphe Nassif, U.S. West
Arie Segev, U.C. Berkeley
Eric Simon, INRIA
Sal Stolfo, Columbia Univ.

2.1 The Questions

I'll first summarize discussions by both the panelists and the participants on the five questions above.

Question 1. *Name an application that will certainly be supported by active database systems in the not-too-distant future.*

- Integrity constraints: Most panelists mentioned this application and the participants unanimously agreed it would be supported. However, there was a question raised: If integrity constraints are all that active database systems will support, then why not simply build special-purpose integrity constraint systems? Most felt the increased generality of active database rules over constraints is worthwhile.
- Simple triggers, alerters, or execution of stored procedures: Many panelists and participants agreed that these applications would be supported. There was some question as to how simple "simple" is (e.g. whether rule interactions are allowed).
- Workflow: Again, many panelists and participants agreed that this application would be supported. There was controversy as to whether active databases are actually the best tool for workflow systems.
- Derived data: This application was not mentioned by any panelist (an oversight?), but it received strong

support from the participants when it was added to the list.

There was some support for the following additional applications: audit trails, index management, planning applications, and maintenance of database statistics.

Question 2. *Name an application that will certainly not be supported by active database systems in the near future.*

- Mission-critical and real-time applications: These applications were mentioned by most panelists, and most participants agreed they would be not supported soon. Examples included air traffic control, power plants, and financial applications.

Other applications discussed as infeasible included: geographic information systems, simulation, data mining, and "anything remotely complex" (e.g. more than 7 rules or more than 5 layers of rule triggering).

In general, participants did not feel it was hopeless to support these applications, but they did feel that support is still a long way off.

Question 3. *Name an area of active database systems in which you are not working but that is crucial to meet the needs of applications.*

- Design, debugging, and analysis tools: Almost all of the panelists mentioned this area, and the participants agreed unanimously. The sentiment was that researchers are doing a good job developing the necessary formalisms, but the formalisms are not being put into practice.
- Performance: About half of the panelists mentioned performance, and many of the participants felt this is one of the most important topics to be addressing. Also mentioned, and related to performance, were benchmarking, and optimizations for constraints.
- "Something that works:" A few of the panelists and many of the participants agreed that building working active database prototypes is crucial. It was pointed out that a real commercial active database system (more than just SQL triggers) is needed.
- Language features: A couple of panelists and a few participants felt that additional language features are necessary. Some supported this claim by suggesting that there may be new language features necessary for certain applications. A stronger argument was that higher-level languages or language features are needed for usability. The need for more language work was controversial; see Question 4.

Question 4. *Name an area of active database systems that is not on the critical path to supporting applications.*

- More work in rule languages: There was considerable controversy here; see Question 3. The panelists and participants finally settled on the fact that we do not need "yet another rule language syntax without an underlying implementation (or semantics)." There was discussion on the difficulty of publishing papers that describe an implementation, which results in many "yet another language" papers and few "underlying implementation" papers.
- Theory: Here too there was controversy (as to be expected). The panelists and participants finally settled on the fact that we do not need more "theory for tenure's sake."
- Toy problems, toy examples, toy improvements: These areas were mentioned by only one panelist but agreed upon by all panelists and participants.
- Deductive databases recast as active databases: Most felt that the two paradigms should be kept separate. There was some resentment about the handful of deductive database papers in the workshop, with an implication that deductive databases really aren't relevant to active databases, especially as pertaining to applications. Not everyone agreed.

There was a general observation that we have the following dichotomy:

complexity of languages proposed:
————————————————————————>

complexity of languages implemented:
——>

It was suggested that we try to, at the very least, ensure that the gap is not increasing, and preferably we should cause the gap to shrink. The conclusion seemed to be that the "complexity of languages proposed" arrow shouldn't grow, allowing the "complexity of languages implemented" arrow to catch up.

Question 5. *Name an area of active database systems that should have been discussed in the course of the workshop but was not.*

There was little disagreement here about a number of neglected topics. The discussion enlarged at times beyond the scope of the question. Topics raised were:

- Implementation issues, complete prototypes, stable platforms, recovery, performance, benchmarking.
- Commercial vendors, standards (although the SQL3 standard was discussed in one talk and additionally during the panel), and real applications.
- Active database technology as it pertains to distributed databases. This issue was not brought up by any of the panelists, but when it was mentioned by a participant everyone agreed on its importance.
- Interaction between active databases and applications.
- Active databases as "glue" for complex applications, especially those requiring interoperability.
- Whether active database systems really provide anything more than simple triggers plus applications (many felt that they do).
- The relationship between transactions and triggers (although a handful of participants felt this was not

so important). A related topic mentioned was process modeling.

- Success and failure stories, experiences by third parties.

2.2 Other Discussion

After discussing the five questions, the panel turned to more general issues. There was some very broad discussion of theory versus practice that followed the usual lines. There was similar broad discussion of the usefulness but difficulty of building complete prototype systems. It was suggested that the active database community should band together to share tools. One suggestion was to use the SIGMOD server on world-wide-web to advertise available software. Some participants felt they would be happy with just a repository of papers.[1]

There was more in-depth and specific discussion of two issues:

- Characteristics of active database rule applications and their managability
- The future of active database systems in light of the emerging SQL3 standard

I'll now summarize the discussion on these two points.

Characteristics of active database applications

Many participants felt that there were two classes of applications. The term "broad and shallow" was coined to describe applications that may have a large number of rules, but the rules don't interact very much. It was felt that these applications are already quite managable. However, it was felt that "deep" applications, where rules have significant interactions, are very difficult to develop and manage, even with very small numbers of rules. Participants felt that we don't have a lot of experience with deep applications. Various participants described rule applications they were aware of or involved in. Most of these applications seemed to fall into the "broad and shallow" category, were not actual database applications, or both.

The future of active database systems

The following questions were posed: Is the SQL3 standard the future of active database systems? If not, what is? It was generally felt that the SQL3 standard is not the future of active database systems, although SQL3 triggers might be used to support more powerful active database systems and/or active database applications. During the discussion, three potential architectures emerged:

1. Active capabilities layered on a traditional DBMS: In this architecture, there is an SQL3 DBMS with triggers at the "bottom," there are one or more components built on top of the DBMS that provide sophisticated active database system capabilities (languages and tools), and finally the applications are layered on top of the active capabilities. It was suggested that this is the architecture we'll see in practice.
2. Integrated active DBMS: In this architecture, the complete functionality of a sophisticated active DBMS is provided by the base system. Applications are built on top of this system. It was suggested that this is the architecture we'll see (and in fact are seeing) in research prototypes.
3. Active object-oriented DBMS. There was consensus among the participants that it is probably impossible to significantly influence the SQL3 standard. That is, standards for relational databases (and therefore most relational DBMS products) will not support sophisticated active database rules. However, it was thought that it may be possible to influence the active capabilities that will be supported in commercial object-oriented database systems, since object-oriented DBMS's are newer and the vendors are more flexible. Such influence might produce an integrated architecture as in architecture 2 above, but for object-oriented active database systems only.

[1]The *ACTNET* community—a consortium of European active database researchers—has since agreed to coordinate at least an ftp server for active database related information.

3 Conclusions

The panel was declared a success since the discussion went well over the allotted time yet almost no one got up and left.

Acknowledgments

Thanks to RIDE-ADS '94 general chairs Lois Delcambre and Eric Simon, and to my program co-chair Sharma Chakravarthy, for putting together a stimulating and successful workshop.

References

[1] U. Dayal, E.N. Hanson, and J. Widom. Active database systems. In W. Kim, editor, *Modern Database Systems: The Object Model, Interoperability, and Beyond*. ACM Press, New York, 1994.

[2] U. Dayal and J. Widom. Active database systems. In *ACM SIGMOD International Conference on Management of Data* (tutorial), San Diego, California, June 1992.

[3] E.N. Hanson and J. Widom. An overview of production rules in database systems. *The Knowledge Engineering Review*, 8(2):121–143, June 1993.