



<http://www.devx.com>

Printed from <http://www.devx.com/Java/Article/21848>

## Author of FuzzyJess Talks About AI and Java

**Bob Orchard is the author of FuzzyJess, the Java-based, fuzzy logic API extension to Jess, and a veteran of many expert systems projects. In this interview, he discusses the state of artificial intelligence (AI), expert systems (ES), and the richness of possibilities for Java developers to utilize his tools for building fuzzy rule-based expert systems.**

by Jason Morris

Over May 17-20, 2004, I had the pleasure of attending the 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE 2004) in Ottawa, Ontario, Canada. I was a guest of Robert (Bob) Orchard, Group Leader of the Integrated Reasoning Group at the Institute for Information Technology - National Research Council of Canada. Bob is the author of [FuzzyJess](#), the Java-based, fuzzy logic API extension to [Jess](#), and a veteran of many expert systems projects.

[Fuzzy logic](#), which models systems with imprecise or vague data, is used in many modern control, financial, and decision support systems. Fuzzy rules are often used to model complex business logic and business decision scenarios. Tools like FuzzyJ and FuzzyJess enable Java programmers to integrate fuzzy-rules into their applications.

The conference gave me a unique opportunity to discuss with Bob many interesting observations about the state of artificial intelligence (AI), expert systems (ES), and the richness of possibilities for Java developers to utilize his tools for building fuzzy rule-based expert systems.

### FuzzyJess and Java

**Jason Morris:** What drew you, a classically educated mathematician, to study fuzzy logic?

**Robert Orchard:** In the late-1980s, a visiting researcher from Poland with considerable expertise in fuzzy logic (Zenon Sosnowski), spent a couple of years in our lab. He added an initial set of extensions to [CLIPS](#) allowing fuzzy reasoning to be done. Just prior to leaving, he gave some talks on fuzzy logic and described the modifications to CLIPS. I started working with him, and when he left, I continued the development of FuzzyCLIPS, modifying the approach and adding many enhancements. We also developed a simple system to control the output of a shower with no knowledge—except the output temperature and flow. This ability to build a control system without a complex set of mathematical equations, using a set of simple English-like rules intrigued me.



**Figure 1.** Bob Orchard is the Group Leader for the Integrated Reasoning Group at the National Research Council of Canada's Institute for Information Technology.

**JM:** So, adding fuzzy logic to rule-based systems seems natural enough. What can you tell us about the original FuzzyCLIPS. Is it still used?

**RO:** Adding fuzziness to a rule based system like CLIPS certainly seemed natural to me. I don't think its impact on AI was huge, but I will claim that FuzzyCLIPS has had an impact. It has been extremely useful in training people about fuzzy rule-based systems since it has been used in many AI courses at universities and colleges. It has been the basis of many research projects (one paper in IEA/AIE 2004 references FuzzyCLIPS), and it has been used in a few commercial applications (some in the US). Since I started tracking about eight years ago, there have been more than 18,000 downloads and 3,000 of those were in the last year.

**JM:** According to Jess creator, [Dr. Friedman-Hill](#), the original Jess project had to do with a novel approach to information protection and network intrusion detection. What was the impetus for FuzzyCLIPS and subsequently FuzzyJ and

FuzzyJess? What were their intended purposes?

**RO:** We recognized that AI would play a major role in the future, and at the time the FuzzyCLIPS project started, we felt a need to be aware of the many facets of AI (machine learning, expert systems, natural language, etc.) including fuzzy logic. Initially we approached groups in the resource industries, trying to interest them in fuzzy logic for control, but they were not ready at the time to absorb this new and—to them—unproven, technology.

We then decided to release the software (FuzzyCLIPS) free for educational and research use, distributing it over the Internet. As our group grew, our expertise expanded in expert systems and machine learning, and the work on fuzzy systems lessened. Because of the strong interest in FuzzyCLIPS (indicated by large numbers of downloads), the growth of Java and the emergence of Jess, I was able to convince myself that a similar capability in Java was worthwhile. I designed the Java classes for FuzzyJ, and with the help of an excellent summer student, an implementation came about quite quickly. A short time later, FuzzyJess was added and the port from FuzzyCLIPS was completed. What we accomplished over this time was the distribution of a couple of tools that have proven useful for students, teachers, researchers and industry.

**JM:** How did the collaboration between you and Dr. Friedman-Hill begin on the FuzzyJess project?

**RO:** The collaboration began some time after I discovered Jess and noted how useful it would be to have a capability similar to FuzzyCLIPS in Jess. I monitored the [Jess User Group](#) for a while and noted how super the support from Ernest Friedman-Hill was. He seemed open to collaborations (encouraging such activities), so I approached him via email in mid-December 1998 to work on a way to integrate the FuzzyJ capabilities (not yet implemented) into Jess in a non-intrusive way. By June of 1999, we were working out the final details for the integration of FuzzyJ (now implemented) and Jess, and the alpha version of FuzzyJess was available in the summer of 1999 with a released version late in the fall of 1999. It was a very fulfilling collaboration, and I think we arrived at a very satisfactory solution. Later I was able to contribute part of a chapter to Ernest's book, "[Jess in Action](#)" (2003, Manning Publications).

**JM:** Describe for the readers what types of problems lend themselves to using fuzzy logic instead of the more traditional [Boolean logic](#).

**RO:** Much of the reasoning that humans do and the way we solve problems is closer to a fuzzy approach than a Boolean approach. The concepts we deal with on a day-to-day basis are most often fuzzy or imprecise (*warm water*, *high pressure*, *strong winds*) rather than crisp or precise (the light is on, the water temperature is *greater than* 212 degrees). So problems that inherently have some degree of imprecision or uncertainty in them can often be solved better using fuzzy logic than Boolean logic or even complex mathematical equations that try to model the uncertainty in the system about which we are reasoning.

Fuzzy logic has been applied in many areas, and its use is still growing. You'll find it in many control applications such as cameras, trains, anti-lock braking systems, elevators, cruise control and washing machines. You'll also find it used in decision making for financial systems (deciding who to give loans to and how much can be tolerated, etc.).

Fuzzy systems are often easier to design, build and maintain and can more easily reflect the human intuition/expertise than can be done with other modeling methods. Fuzzy systems can deal with imprecise and missing information, can model complex non-linear systems, can be written in natural language, are robust to changing situations, and they often out-perform mathematical systems and even humans in control and decision making. However, one would be wise to be careful to note when not to use fuzzy logic as well. (See the Mathworks Fuzzy Toolbox "Getting Started" documentation for a good explanation.)

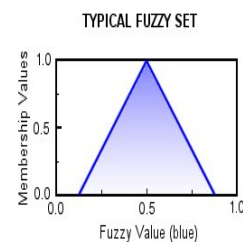
**JM:** OK, but how is this different from using ordinary probability and statistics to reason with uncertainty?

**RO:** There is great controversy between the fuzzy camps and the probability camps over the relationship between probability theory and fuzzy sets. In my view of things, membership in a *fuzzy set* is not a probability in general.

[Didier Dubois](#) and [Henri Prade](#) are two of the pioneers of the theory and application of fuzzy systems. They wrote an article that dealt with this controversy ("Fuzzy Sets And Probability: Misunderstanding, Bridges And Gaps, 1993, Proceedings of the Second IEEE Conference on Fuzzy Systems"). Others have stated that probability deals with the *likelihood* that something has a particular property, whereas fuzzy logic deals with the *degree* to which the property is present.

**JM:** With what additional technologies or academic fields should Java programmers be familiar before attempting to use FuzzyJess?

**RO:** Java programmers ought to become familiar with rule-based systems and at least the rudiments of fuzzy logic before attempting to use FuzzyJess. It can take some time for programmers, used to a language like Java, to understand the operation of a



**Figure 2. A Typical Fuzzy Set:** Fuzzy sets measure how much a value "belongs" to the set. In this case, as the fuzzy value approaches 0.5, you say that its membership approaches 100 percent.

rule-based system like Jess. *Jess in Action* can help with both of these, but some primers on fuzzy logic might also be required.

**JM:** What is the difference between Fuzzy Jess and Jess proper? How does FuzzyJess work?

**RO:** FuzzyJess is just Jess with some extra capabilities for handling rules with fuzzy pattern slots and that assert facts with fuzzy slot values. However, one has to learn the language of fuzzy systems as implemented in FuzzyJ/FuzzyJess. This includes: FuzzyVariables and FuzzySets that allow you to define the concepts that you will be describing, like temperature with the terms hot, cold, warm, and cool; FuzzyValues that are specific instances of the fuzzy concepts, like cold or hot temperature; and FuzzyRules.

So, a non-fuzzy Jess rule might look something like this:

```
(defrule crisp-temperature-rule
  (air-temperature ?t(> ?t 85))
=>
  (assert (increase-fan-speed 200)))
```

While a fuzzy Jess rule might look something like this:

```
(defrule fuzzy-temperature-hot-rule
  (air-temperature ?t&:(fuzzy-match ?t "hot"))
=>
  (assert
  (increase-fan-speed (new FuzzyValue ?*speedFvar* "large"))))
```

**Author's Note:** If you are wondering what this code is, because it certainly isn't Java, it is Jess's internal scripting language also referred to as "Jess." Its syntax is similar to CLIPS, but it differs in functionality and implementation since it primarily serves to expose Jess's Java API in an interpreted, command-line way. In other words, you can program in Jess (and hence FuzzyJess) by writing Jess batch files, by programming its Java API directly, or by a mixture of both! This flexibility is one of the hallmarks of using Jess.

The fuzzy-match function compares a fuzzy value in the fact slot (?t) to a fuzzy value defined in the 2nd parameter of the fuzzy-match function ("hot" temperature). If there is a match, the rule fires and an increase-fan-speed fact is asserted with a fuzzy slot describing the change—in this case the change of fan speed will be "large."

You need to understand how the fuzzy outputs from the rules are combined automatically and how to defuzzify (convert from a fuzzy to a crisp value) the final fuzzy output values. Of course there are plenty of details to attend to, but this is described in the documentation for the FuzzyJ Toolkit and FuzzyJess as well as in the book, "[Jess in Action](#)," by Ernest Friedman-Hill (2003, Manning Publications).

**JM:** Now, rule portability and system scalability are major issues in constructing enterprise-class, rule-based expert systems. Can FuzzyJess rules be constructed in other formats like XML, RuleML, or DAML?

**RO:** Because we don't change the syntax or semantics of Jess in any way, the things that can be done with Jess can be done with FuzzyJess. Something like RuleML is crippled because it tries to meet the needs of all rule based systems. Therefore, the rule systems one can express in RuleML are a very limited set of what can be expressed in Jess. You might be able to represent a FuzzyJess rule system in RuleML, but it wouldn't execute on any runtime rule engine except Jess with the FuzzyJess additions. RuleML will likely never allow one to define fuzzy rules.

**JM:** What can you mention about the companies have licensed FuzzyJess and how they are using it in industry?

**RO:** As a federal research lab and not a commercial organization, we are not putting a lot of effort into selling FuzzyJ/FuzzyJess on a mass-market basis. However, we do require that commercial users of FuzzyJ/FuzzyJess purchase a license. I am not able to reveal the names of the companies (there are only a handful), and they have not told us what products they have marketed or how they are using to toolkit. I can tell you that all of the companies are US companies. Perhaps this says that US companies are some of the most honest in the world in honoring on-line agreements.

**JM:** The next major release of Jess (version 7.0 code named Charlemagne) will be an Eclipse plug-in. Will there be any noticeable changes or upgrades in FuzzyJess (like a fuzzy rules editor) to coincide with its release?

**RO:** FuzzyRules are just like Jess rules, so the rule editor for Jess will handle fuzzy rules by default. However, what many other commercial systems have that is quite nice is an interface for defining fuzzy sets and terms and even the fuzzy rules graphically. This won't likely happen for FuzzyJess in the near future. Unless there are specific user requests, I expect that for the next while there will only be minor updates to the FuzzyJ/FuzzyJess Toolkit (the users appear to be satisfied with the current functionality).

**JM:** What features of FuzzyJess make it a better choice for programming fuzzy rules than something like [FIDE](#) from Apronix

or the [MATLAB Fuzzy Logic Toolbox](#) from the Mathworks?

**RO:** Well, both FIDE and the MATLAB Fuzzy Logic Toolbox are excellent. FIDE in particular provides a nice graphical environment for creating fuzzy systems and testing them. However, FuzzyJ/FuzzyJess is, I believe, more flexible since it is a Java API. It can be used in any Java program, and when used with Jess, it gains the full flexibility of the Jess rule-based environment (fuzzy and non-fuzzy reasoning with a rich pattern matching capability in the rules). It is relatively simple for end-users to extend due to the strong Java/object connection. For example, one can easily add one's own fuzzy set types or defuzzification methods or even change some aspects of the way the fuzzy rules combine their results.

## Transferring AI Technology

**JM:** Let's shift to one of my pet topics. The problem of transferring AI/ES technology from academia and other research institutions such as NRC to the manufacturing and service sectors was continually discussed at the conference. What has been your experience and what did you observe?

**RO:** I'm not sure that the problem is restricted to AI technologies being transferred. What I've seen is a reluctance of companies to accept anything that they don't understand or that has not proven itself. Many companies appear to be late adopters and unwilling to put even a relatively small effort into testing new technology.

In some cases, we've asked for access to their data and access to a small amount of time from domain experts so we can show how a technology might help. Even if we get access to the data, access to expertise is difficult (these are some of the companies most valuable people). Sometimes we find that the company is not collecting all or the right data that would lead to success, and it is a challenge to change the process to fix this, especially without a company "champion." *Now, none of this is new to people who are trying to transfer technology or produce real world products.*

One needs to be persistent, to be upfront with the clients about the expectations of the technology and their need to be involved, to use real data with all of its deficiencies, and to be willing to go beyond just a simple prototype. The reality is that a great deal of AI technology has been adopted and will be adopted in the future.

**JM:** Developers and development managers are often at the cutting edge of technologies that they would like to bring to bear on a particular problem, but they must defend their recommendations with some estimate of the ROI to convince senior management to adopt their plans. What lessons and words of advice can you share about technology transfer processes? What strategies worked, what didn't, and what can programmers, team leads, and architects do to "sell" technologies like FuzzyJess as components in solutions?

**RO:** The answer to this question sounds like the makings of a good book! However, I have had some experience in transferring technology, and I don't think that there is a formula that works [all the time]. Each company is different, and each technology is different, but here are some things that I might offer:

- Convince the company to do a study where you demonstrate the benefits to them.
- Keep the cost to the company as low as possible, but don't do it without getting commitment to the project and all of the support you need (access to any real data, to company expertise, and to those who will be most affected by the technology).
- Make sure that everyone is very clear about what the expectations are for the new technology: what it will do and what it won't do; what the risks are that could impede its success.

## The Future of AI

**JM:** As the technology has matured, the number of AI and ES success stories has grown. Yet the ratio of attendees from industry to academia at IEA/AIE this year was low. Why do you think that is?

**RO:** The low ratio of industry attendees (and authors) to academic participants may, I suspect, be related to a couple of things.

First, the IEA/AIE conference is very general in nature, covering a wide range of AI topics. Industrial participants may prefer to attend conferences more specific to their domain (e.g. telecommunications), and there [they may] see AI applications as one of many solutions to their problems.

Second, although the papers in general deal with *applications* of AI, they do have a research component, and this still seems to be driven by academics who are required to publish. In addition, most of the reviewers that we are able to recruit are from academia, and despite the emphasis on applications, they do tend to want to see papers that also have a research component.

**JM:** Judging from the attendance at IEA/AIE 2004, it seemed that the US is under represented in the AI/ES field. Can this be

true?

**RO:** No, I don't think that the United States was under represented or that it's cause for alarm.

Considering that a couple of the most prestigious conferences in AI are in the United States, there is clearly a strong contingent of world-class US researchers who have and still are greatly influencing the progress of AI. For example, the [KDD](#) (Knowledge Discovery and Data Mining) is probably the premier conference for data mining in the world, and the [AAAI](#) is another American-based organization held in high regard around the world.

**JM:** Let's consider Java for a minute. At IEA/AIE 2004, I saw indications that Java is becoming the de facto standard for implementing many AI/ES solutions. What is your opinion?

**RO:** I'm not sure that Java is becoming a de facto standard for building AI solutions [yet]. However, I believe that it is true that many AI components are being implemented in or ported to the Java platform. There are expert system tools like Jess that make integrating a rule-based component of a system with a Java application very simple and powerful. There are toolkits like those for data mining that have been implemented in Java (see [Weka](#), a collection of machine learning algorithms for data mining or the [JSR 73: Data Mining API](#)). Consider the Open Source Project Joone, the Java Object Oriented Neural Engine.

The point is that, given a growing set of AI-related APIs in Java and the current tendency to teach Java as a first programming language, it probably makes sense to use the Java platform in AI courses. In our group at NRC, much of the software that we develop is done with Java. However, we also use C/C++ and other tools as suits the needs of the project or the preferences/skills of the developer. I personally believe that Java has a strong AI presence and it will continue to grow.

**JM:** Good! So your final impression of the state of the worldwide AI/ES field, given the IEA/AIE 2004 presentation topics and attendance, was positive?

**RO:** Yes, the attendance at IEA/AIE 2004 was up by about 20 percent over previous years. This is encouraging and suggests that the field is strong. More importantly, perhaps, is the evidence that AI technologies are being applied in a wide range of application areas. Data mining, distributed intelligent systems, and soft computing (fuzzy logic, neural networks, genetic algorithms, etc.) are particularly well represented. Applications in bioinformatics, for example, will be key in the search for knowledge about gene and protein function.

**JM:** If a friend had \$10,000 to invest in an emerging technology and asked your opinion, what would you recommend and why?

**RO:** Two areas come to mind. One is nanotechnology and the other is bioinformatics. Both are making progress, and I think the future looks good. More specifically on bioinformatics, I see that tools for data mining and text mining in genetics and proteomics research are critical to the ability of biologists and other scientists to make and validate discoveries. There is still a lot to be discovered, and there is great potential for wealth generation.

Thus, it appears that artificial intelligence and expert systems are alive and well in the US as well as abroad, and Java technology is playing a growing role in their adoption. Java is increasingly used as the language of choice for introductory programming classes, and many research applications are now either originally written in Java or are being ported to the Java platform (for some applications, see "[Innovations in Applied Artificial Intelligence: Proceedings of the 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems](#)," by B. Orchard, et. al. (2004, Springer Verlag).

Though Java programmers are intimately familiar with programming bivalent (true/false) logic, it is difficult to create intelligent software that reasons with imprecise or uncertain data—data that that isn't necessarily "black or white" in the logical sense but rather some shade of gray in between. Fuzzy Logic, which is a branch of mathematics that describes a calculus for this kind of data, can be used to model imprecise systems. FuzzyJess is a robust Java API extension to Jess that allows Java programmers to write rule-based expert systems using fuzzy logic.

**Jason Morris** has been involved in software development since 1993. He runs Morris Technical Solutions, a consulting firm that specializes in engineering information technology.