Table 2: Questions to Ask about an Implemented System

Why, Why, Why, ...

- What was the motivation for building the system? If there is more than one motivation, can you prioritize them?
- Did you consider building on or adapting another system as an alternative to building one from scratch? If so, which other systems did you consider, why did you consider each one, and what was your conclusion in each case?

What's It Good For?

- Is the system intended to be used for some application? If so, can you characterize the kinds of problems it is intended to be used for independently of the system?
- Is there a target user community? What has been your interaction with that community?

How Does It Measure Up?

- Generally speaking, to what degree do you feel the system has achieved the goals that motivated it?
- What properties (if any) of the system do you measure or formally evaluate in any way? For example, do you have some measure of system performance?
- Do you have some standard examples that you use in working on the system? How did you come up with them? How do you use them? Would they be appropriate for other systems? If not, why not?
- What other systems does it make sense to compare the system with and, in each case, what kind of comparison is appropriate?

Models and Predictions

- Do you use any formal models of either the system behavior or the problems it solves? If so, please describe them briefly.
- Can you predict, using one of the models above, how certain design changes will affect the behavior of the system? If so, please give a few examples.
- Can you predict, using one of the models above, how system performance will be affected by certain variations in the problem space? If so, please give a few examples.
- Can you identify one or more algorithms that are central to the operation of the system? If so, please describe them briefly. Are any of these algorithms novel?

Report from the Trenches

- What parts or aspects of the system have consumed the most implementation resources?
- Are there parts of the system implementation that failed to behave properly and were subsequently discarded (or kept)? If so, describe them briefly along with why they failed.
- What parts of the system needed to be optimized? Why? How did you find out?
- Did you explicitly explore engineering tradeoffs in any parts of the system design, such as time versus space? If so, describe them briefly.
- What implementation foundation (e.g., programming language) did you choose? Why? Did you consider alternatives and choose based on some technical differentiation (if so, what was it)?

Lessons Learned

- What aspects of the system are you most proud of and would you insist on doing just the same way the next time?
- Do you have any advice for those building similar systems regarding what *not* to do? What design decisions would you make differently? (Try to rank these in order of importance and impact.)

Publish or Perish

- Have you published any papers about the system? If so, in what conferences and/or journals? If not, have you submitted papers and had them rejected? If so, to what conferences and/or journals? Why were they rejected?
- Can you cite any published papers in AI or elsewhere in computer science that you feel are role models for how to write a good paper about an implemented knowledge representation and reasoning system? Please suggest both conference and journal papers (if you know of any).
- What types or groups of people do you think would or should be interested in the system, and why? In particular, what benefit do you think the system could be to people other than those who might actually use it?