To be published in:
**Electronic News Journal on Reasoning about Actions and Change**
Vol. 1: [*pages tbd* ], 31.10.1997
http://www.ida.liu.se/ext/etai/rac/notes/1997/06/

# Modelling Language vs. Repository of Common-Sense Facts

## Two Perspectives on Knowledge Representation

*David Poole*
*University of British Columbia*

My guess is that we are in a phase of normal science. The revolution is coming. When we have to explicitly consider uncertainty much of what we think we understand now will have to be thrown out.

In order to go about evalation, we have to make our goals clear. (If it doesn't matter where you want to get to, it doesn't matter much which way you go, to paraphrase Lewis Carrol). There are two quite different goals people have in building KR system; there is much confusion generated by not making it clear what you are doing (so much so that the researchers who take one view often don't understand what the others are doing and why). These are:

1. A knowledge representation as a modelling language. If you have a domain in your head you can use the KR to represent that domain. The builder of a KR is expected to give a user manual on how to axiomatize the domain. There are right ways of saying something and there may be wrong ways of saying it. Missing knowledge may mean something. Prolog and Bayesian networks are examples of such knowledge representations.

2. A knowledge representation as a repository of facts for commonsense reasoning. Under this scenario, you assume you are given a knowledge base and you are to make as much sense out of it as possible. It isn't OK for the designer of the KR to prescribe how a domain should be axiomatized. The KR should be able to get by with whatever knowledge it has. Much of the nonmon work assumes this (as far as I can see).

If you goal is the first, you probably want a very lean language which doesn't provide multiple ways of doing the same thing. You want to provide a recipe book about how to go about modelling a domain. It should be judged by whether someone can go from an informal problem (not a representation of a problem) to a solution efficiently. Does it provide a good way to think about the world? Can it exploit any structure of the domain for efficiency?

If your goal is the second, you probably want a rich language that lets you state as much as possible. It should be some free-form language that doesn't constrain you very much. Here we need to go from a representation of a problem into a solution. Does it provide reasonable answers? Can the user debug the knowledge base if an aswer is wrong?

I have two ways of judging a representation:

1. Can I teach it to cynical undergraduates without squirming? Can I make a case that this is the obvious answer?

2. How well does it work in practice? What is the range of practical problems for which it provides a solution?