women already established in AI today were attracted to the subject before computers acquired their macho image. At least they had not been put off using computers whilst at school.

It is too early to say whether this scenario is actually the case. What we can say is that the image of computing has reached a very low ebb amongst girl school-leavers and that this will without a doubt have a knock-on effect on the number of people choosing computing or computer-related careers. This must have an effect on the AI community if only in the fact that the pool of possible researchers will be smaller. At worst, it will starve AI research of the particular skills that women can offer. There are a number of publicity and research initiatives beginning to emerge this year in order to find both short-term and long-term solutions to the problem. These will involve finding reasons why girls are dissuaded from careers in computing, and finding ways to attract them to the subject. Ironically, one way might be to increase pupils' awareness of AI and AI techniques because this is one of the aspects of computing that is most likely to appeal to girls.

REFERENCES

1 Lovegrove, G.L. and W. Hall (1987). Where Have All the Girls Gone?, *University Computing*. December, 207-210.

Humpty Dumpty, Private Languages and Logic Programmers

T J M Bench-Capon
Department of Computer Science, University of Liverpool

In his stimulating article, 'Pathologies of AI: Responsible Use of AI and Professional Work', in *AI & Society* (1988) 2.1, Ronald Stamper criticises some of the dangerous semantic habits of logic programmers. In short he claims that logic programmers, through their use of fat predicates, play Humpty Dumpty and invent a number of private languages, and in so doing exacerbate the difficulties of 'incorporating language into an artificial system'. This is a fairly common type of criticism of some aspects of logic programming, and there are a number of issues here which merit further discussion.

It is perfectly true that logic programmers have a tendency to use long predicate names which look 'deceptively like natural language' and that, as a result of this, naïve users might be misled into thinking that the programs are using language intelligently by forming these phrases from their constituent words. This in turn might lead such users to ascribe intelligence to the program where none exists. But it is not right to ascribe to the programmers an intention to deceive such users; their aim is to make it as easy as possible for the users to understand the output of the program, and this is very much facilitated by the use of these lengthy predicate names.

Humpty Dumpty is much maligned; when he says 'when I use a word it means just what I choose it to mean — neither more nor less' he is putting forward a perfectly respectable philosophical position, provided he chooses what his words will mean in

advance of using them, can explain what he has chosen them to mean to others, and is consistent in his use. Humpty Dumpty does this; he teaches Alice what he means by 'glory' and 'impenetrability', and indeed philosophers have triumphantly crowed 'there's glory for you!' at one another ever since. All this is quite different from the logically private languages disposed of by Wittgenstein in the *Philosophical Investigations*. Those private languages were impossible, not because words were assigned meanings by their users, but because of the nature of the meanings assigned. Because the denotation of the terms of these private languages were sensations they were necessarily incommunicable to others, precluding the possibility of use by others, and thus rendering them useless as means of communication. Worse still, if we accept Wittgenstein's arguments, the lack of objective criteria for the use of such words renders them useless even for internal monologues.

The idiolects ascribed by Stamper to myself and Sergot may be personal in the Humpty Dumpty sense, but they are not private in the sense criticised by Wittgenstein. And indeed they are not even so personal as to be incapable of being immediately understood as equivalent by competent English speakers. People are surely allowed their own personal prose styles without ceasing to speak the same language. Just as when we write English we have our own ways of expressing ourselves, so too we should expect these stylistic differences in the choice of predicate names. This does not prevent us from asserting the same propositions, nor from understanding one another. This interpretation does mean that the relations between the propositions expressed in the clauses of the logic program must be true on the ordinary English interpretation of the words involved. If they were genuinely symbols in a private language it would not be open to a critic of the program to reject clauses as false, but in the cases cited this sort of criticism is applicable. The clauses of the programs are meant to be true statements expressing relations to be found between concepts readily expressible in English, not stipulative definitions. It is also worth noting that it is not 'assumed that meanings are completely specified by the axioms of the system'. For some predicates will be undefined within the program, and the systems under discussion will ask the user to say whether these predicates are satisfied or not. Thus the user determines the interpretation of such predicates, and the only way in which the programmer can hope to harmonise his interpretation with that of the user is to rely on a common interpretation derived from a shared competence in English.

The real problem is a radically false model, in which it is an intelligent system which is to communicate with the user. This is simply not even aspired to in the sorts of program discussed here. The correct model is of the programmer communicating with the user through the medium of the program. On this model lengthy phrases no more lose their meaning by being written in a logic program than they do by being written in a book. The test is not whether the program understands the phrase, but whether the user understands the phrase. If the user does not understand, this is, of course, a failure of communication on the part of the programmer, just as a reader's lack of understanding of a piece of prose is a failure on the part of the author. Of course, this means as well that the resulting program is no more intelligent than a book, which is a consequence I at least am more than happy to accept, never having claimed intelligence for programs of this sort, and agreeing fully with Stamper's 'dislike of the ideologically driven search for the machine which will replace human intelligence'.