Bob Kowalski: A Portrait

Marek Sergot
Department of Computing
Imperial College of Science, Technology and Medicine
London SW7 2BZ

Introduction

The hardest part about writing an introductory piece for a celebratory volume such as this is finding the right opening. It has to hit the right tone straight away—affectionate, respectful, but not too sweet and cloying. I had tried and discarded half a dozen attempts when, more in desperation than in any real hope, I turned to technology and typed 'Bob Kowalski' into a WWW search engine. I am not sure what I expected to find. Some previously unpublished tidbit perhaps on which I could build an insightful and original opening. The search yielded a great many results. On page 12 I came across an entry from the newsletter of the *Tulsa Thunder*, a girls' football ('soccer') team in the US. According to one person quoted there: "Bob Kowalski was one of the first influential coaches I had. He was an all-round good guy." I was about to discard this interesting observation (it is a different Bob Kowalski) when it occurred to me that in fact this quotation would serve perfectly as an opening for this piece. I had wanted to begin with remarks about Bob's inspirational influences and what a good guy he is, but could not decide which should come first. Bob has certainly been one of the most influential coaches I ever had, and as the rest of this volume testifies, an inspirational influence on many, many others too. He is an influential and inspirational coach, and he is an all-round good guy.

The 'all-round good guy' part was particularly tricky to introduce. How does one bring this up? For now I will just state it as an assertion, and leave the reasons to emerge in the course of the article.

The editors encouraged me to give this introduction a personal tone, and so at this point I display my credentials. Among the many important and long-lasting contributions Bob Kowalski has made to the development of Computer Science, a lesser known one is that he is the main reason I decided to stick with Computer Science myself. In the Spring of 1975 I was halfway through an MSc course in Computer Science at Imperial College. I was disillusioned and disappointed and bored. I could not believe there was so little in it. It was like plumbing, but without the intellectual challenge. I turned up for a research seminar by Bob who had just moved to the Department of Computing (or Computing and Control as it was then called) from Edinburgh. Like many others before me and since, I was inspired—inspired by the prospects of new and exotic applications, a little, but more by the enthusiasm and energy of the speaker, and most of all, by the elegance of the logic programming story that he unfolded before us. There was something in computing after all.

Since then I have had the fortune to work closely with Bob, briefly in the summer of 1975, and then more or less continuously since 1979, in close collaborations throughout the 1980s and early 1990s, and then more at a distance as our interests diverged.

The account of Bob's life and work given here is based on my memory of Bob's musings and recollections in casual conversations over the years. Many of our colleagues would recognise these recollections, I am sure. I tried to fill the gaps by conducting subtle interrogations of Bob on the last few occasions I have had the opportunity to chat with him. These interrogations were so subtle that he did not notice and they failed to yield anything at all. By luck, just as this volume was going to press, Bob distributed to a few of us a short autobiographical piece he had written in response to some request or other he had received from a student. I was thereby able to confirm the facts as I had remembered them. I have also taken the liberty of lifting three small quotations from Bob's own version, where I had remembered the gist of what he had said, but where his own words have a particular interest.

I should say that Bob has not had the chance of reviewing this manuscript before it went to press. There may be mistakes in points of detail. Moreover, the opinions expressed are mine, and not necessarily the same as Bob's.

Some biographical details

Robert Anthony Kowalski was born on 15 May 1941 in Bridgeport, Connecticut. He has two younger brothers, Bill and Dan. His father was the son of Polish immigrants to the US; his mother, if I recall correctly, came to the US from Poland as a young girl. Although his parents would speak Polish occasionally at home, the boys did not. Bob attended a Catholic primary school attached to the Polish parish and then—much more significantly—a Jesuit High School. This had a lasting influence, clearly, since Bob mentions it often. I was most impressed when I discovered it, because I was educated by another brand of Catholic brotherhood, not nearly so famous, and the products of a Jesuit education have always held a certain cachet for me. Jesuit schools got prominent mentions in our History books. When I think of Jesuit schools in the USA in the 1950s and 1960s I immediately get a mental image of something like the jet fighter-pilot training school in the film *Top Gun* but with intellectual missiles instead of heat-seeking ones. By coincidence, there was another American Jesuit-educated Professor in the Department of Computing at Imperial College, and so I had an opportunity to try to detect the common features. The results were inconclusive.

Bob says that he was not an academically outstanding pupil at High School, until he discovered, or had discovered in him, an aptitude for Latin, in which he represented the school in contests in New England. I have some difficulty in imagining what a Latin contest in New England must be like, but the important thing is that it awakened Bob's academic ambitions, and encouraged him to undertake independent reading, especially in areas of Philosophy and

the Philosophy of Science which have remained a lifelong interest.

Bob began undergraduate studies in 1958 at the University of Chicago. He enjoyed the academic and intellectual environment. His courses included introductions to mathematical logic. However, other components of the courses were much more tedious and this, together with aspects of the social life, led him to abandon his studies at the University of Chicago early in his second year, in November 1959.

He resumed his undergraduate studies the following academic year, this time in his home town at the University of Bridgeport. He majored in Mathematics. In 1963 he won Woodrow Wilson and National Science Foundation Fellowships for graduate study and was admitted to the PhD programme (in Mathematics) at Stanford University. Jon Barwise was a classmate and a friend. The academic year 1964-1965 was spent on an exchange programme at the Mathematics Institute of the Polish Academy of Sciences and the University of Warsaw, noted for its work in Mathematical Logic. Besides studies of logic, and meeting and visiting his Polish relatives, in that year Bob learned Polish, he met and married his wife, Danusia, a student in the Mathematics Department at the University, and he discovered that the world was not as he had been led to believe it was.

One of the first conversations I remember having with Bob was of his experiences of that year in Poland. A childhood in the US in the 1950s and an education with the Jesuits had painted a clear picture of what life in Poland would be like. He expected that there would be very severe restrictions on personal and other freedoms. What he found was quite different, and in particular that the people seemed to have much more freedom than he had been told to expect. The discrepancy was so great that he felt he had been badly let down and misled—'cheated' was the word he often uses when speaking of it.

On his return to Stanford with Danusia for the academic year 1965 he found it increasingly difficult to focus on studies of mathematics. The war in Vietnam was escalating, and he became active in the protest movement. I knew that he had participated in marches and demonstrations, and he had told me that his specialty had been in generating new ideas for protests. It was only when I read his autobiographical piece as this volume was going to press that I discovered he also participated actively in some of his own schemes. I discovered, for example, that he devised and with a childhood friend from Bridgeport took part in a 'bombing' campaign to drop leaflets from airplanes. The first sortie nearly ended in disaster. The last mission also stands out. In Bob's own words:

> Our main goal was to 'bomb' the Rose Bowl football game in Los Angeles. Ray and I worked out an elaborate scheme to change the registration number on the side of the plane, ripping the false numbers off in mid-flight, to minimise the chance of getting caught when we made our getaway. Unfortunately, when we landed in the Mojave Desert to change the number, the plane burst a tire, and we were too late to get to the Rose Bowl in time for the game. We bombed Disneyland instead.

Bob decided to leave Stanford in the middle of the academic year in 1966, which gave him a Master's degree. Having looked for work, mostly outside the US, he eventually took a position for a year as Assistant Professor and Acting Head of the Mathematics Department at the Inter-American University in San Juan, Puerto Rico. His first daughter, Dania, was born in Puerto Rico during that year.

In 1967 he accepted an IBM Research Fellowship to undertake PhD studies in the Meta-mathematics Unit directed by Bernard Meltzer at the University of Edinburgh. The research topic was the mechanisation of mathematical proofs. Bob was not particularly enthusiastic about the topic, and even less enthusiastic about Computer Science, but was determined to finish his PhD quickly. Of course we now know that he could not have arrived in a new place at a better or more exciting time. Edinburgh was a world-renowned centre of research in Artificial Intelligence and attracted visiting researchers from all over the world. A major influence was that of Alan Robinson, the inventor of resolution, who was spending a year's sabbatical in Edinburgh. Bob wrote his first research paper[1] on some ideas of Robinson's on semantic trees jointly with another new PhD student, Pat Hayes, now a prominent figure in the field of Artificial Intelligence himself of course.

Bob finished his PhD, on studies in the completeness and efficiency of resolution theorem-proving, in just over two years, and then stayed at Edinburgh on a postdoctoral Fellowship. His two other daughters, Tania and Janina, were born in Edinburgh.

The history of the origins of logic programming have been documented by the main participants elsewhere[2] and I make no attempt to reproduce them here. Bob had been working on the SL form of resolution[3] with Donald Kuehner, a former teacher from the University of Bridgeport whom Bob had persuaded to come to Edinburgh to do his PhD. It was becoming clear that the goal-directed nature of SL-resolution provided a procedural as well as a declarative reading for logic clauses, so giving the basis for a new kind of programming language, and a way of reconciling the debates about procedural and declarative representations that were starting to dominate AI research. In the summer of 1971, and then again in 1972, Bob was invited by Alain Colmerauer to visit him in Marseilles to work on the application of SL-resolution to Colmerauer's work on natural language understanding and question answering. These collaborations focussed initially on the applications of clausal logic and SL resolution to grammars and to parsing, but from them emerged many of the principles for the use of logic as

[1] Kowalski, R.A., Hayes, P.J. Semantic trees in automatic theorem-proving. In *Machine Intelligence 4* (B. Meltzer, D. Michie, eds), Edinburgh University Press, 1969, pp181–201. Reprinted in *Anthology of Automated Theorem-Proving Papers*, Vol. 2, Springer-Verlag, 1983, pp217–232.

[2] See e.g. Kowalski, R.A. The Early Years of Logic Programming. *CACM* 31(1):38–43 (1988).

[3] Kowalski, R.A., Kuehner, D. Linear resolution with selection function. *Artificial Intelligence* 2:227–260 (1971). Reprinted in *Anthology of Automated Theorem-Proving Papers*, Vol. 2, Springer-Verlag, 1983, pp542–577.

a progamming language, and led Colmerauer to the design and implementation of the logic programming language Prolog in 1972.

The next few years at Edinburgh were spent developing the new logic programming paradigm and laying down its foundations. Edinburgh provided the perfect environment. There were enthusiastic colleagues, notably Maarten van Emden, with whom he developed the fixpoint semantics[4] and ideas for applications, and David Warren, Bob's first doctoral student, who designed and implemented the 'Edinburgh Prolog' compiler. Bob's hugely influential "Predicate Logic as Programming Language" was published in 1974[5]. There were also visiting researchers from institutions around Europe—Maurice Bruynooghe, Keith Clark, Luis Pereira, Peter Szeredi, Sten Åke Tarnlund, among others—with whom Bob formed lasting collaborations and friendships. He travelled extensively, mostly in Europe, spreading the ideas. He completed a long technical manuscript, later to become the core of his book *Logic for Problem Solving*[6]. He also continued to work in automated theorem proving. His connection graph proof procedure was developed during that period.

In January 1975 Bob left Edinburgh to take up a Readership[7] in the Department of Computing and Control at Imperial College, London (now the Department of Computing). The second half of the 1970's was spent finishing his book, producing other milestone papers, such as his famous *Algorithm = Logic + Control*[8], and building up activity in logic programming at Imperial College. Keith Clark, who had been a visitor at Imperial College when I was first there in 1975, had moved from Queen Mary College in London to a permanent position at Imperial by the time I returned in 1979. Chris Hogger had completed his PhD and although still a member of another Department would shortly join the Department of Computing. A number of other colleagues in the Department had been enticed to work in logic programming. The first Logic Programming Workshop, which eventually evolved into the ICLP series of International Conferences on Logic Programming, was held at Imperial College in 1976. I attended that workshop myself, though what I mainly remember about it was the workshop party that was held at Bob and Danusia's home in Wimbledon one evening, and the rolling tobacco that I was induced to try by Danusia's father. All this talk of logic programming made my head spin (though it might have been the tobacco). I didn't even smoke cigarettes. Natural politeness made me accept.

By 1979, the Logic Programming Group at Imperial College consisted of Bob, Keith Clark, Chris Hogger, two or three other members of staff who were starting to work in the area, and six PhD students and research assistants, of which I was one. Logic programming, in various guises, was part of the curriculum

[4] van Emden, M., Kowalski, R.A. The semantics of predicate logic as a programming language. *JACM* 23(4):733–742 (1976).

[5] *Proceedings of the IFIP Congress, Stockholm*, North Holland, 1974, pp569–574.

[6] North Holland Elsevier, 1979.

[7] A Readership in the UK is a senior academic position, somewhat below the rank of (Full) Professor, and traditionally with an emphasis on research rather than teaching.

[8] *CACM* 22(7):424–436 (1979).

of the undergraduate and MSc courses. There was also an active group in functional programming with whom we had close contacts and regular joint seminars. There was a constant stream of visitors and speakers. My memory of Bob and Danusia's home in Wimbledon will be that there always seemed to be someone staying there—a brother from the USA, a relative from Poland, a former colleague from Edinburgh, a logic programmer passing through. It was not always easy to tell the difference, except that the brother from the USA and the relative from Poland would usually be sanding down floors or painting the kitchen door. Bob was appointed Professor of Computational Logic at Imperial College in 1982.

I realise that I am starting now to conflate Bob's biography with the fortunes of the Logic Programming Group at Imperial College, but for much of the 1980s and 1990s the two are so inextricably linked that it is impossible to disentangle them.

The 1980s saw a massive expansion of the Logic Programming Group, and of Bob's personal standing and celebrity in Computer Science. The group was already growing with the acquisition of a number of new projects and grants when in 1981 came the announcement by MITI in Japan of the Fifth Generation Computer Project. The project aimed to leapfrog a generation of computer system development in 10 years, to a position of dominance over IBM, and to a new era of advanced knowledge processing applications. Logic programming—to widespread surprise—was identified as the core technology. Various governments, including the UK, were invited to participate. Since we at Imperial College were at that time the largest and most active centre of research in logic programming, we expected that we would be playing a substantial role in the Fifth Generation Project, especially if the UK government decided to accept the invitation to participate.

Bob, who was already a very well-known figure in computer science, became something of a celebrity. At the ICLP conference in Marseilles in 1982 I was chatting to him over breakfast when suddenly a camera was thrust between us and he was asked to pose for photographs. He was photographed at lunchtime, and in the afternoon breaks when we all walked down to swim in the sea, his head was photographed again as it bobbed up and down in the Mediterranean swell.

I hesitate to dwell too long on the Fifth Generation Project and the associated politics of the UK's response since much of the account would be second hand. However, these matters dominated the 1980s in one way or another, and accounted for much of Bob's time and energy for nearly a decade. Bob had been working very hard at putting a case to the Science Research Council for what it called a Specially Promoted Programme (SPP) in logic programming. The argument was not just that logic programming was the enabling technology for new AI and 'knowledge processing' applications, but that it provided a unifying foundation for developments in AI, in programming languages, in formal methods for software engineering, and in parallel computing. The case for the SPP went through several iterations but was eventually swallowed up in the UK's

general response to the Fifth Generation Project.

Not everyone in the UK was as enthusiastic about the role of logic programming as the Japanese. The UK government's reaction to the Fifth Generation Project was to set up a committee, chaired by John Alvey, to recommend the best course of action. That committee was advised by another layer of committees drawn from academia and industry. Naturally, most of these advisers saw it as an opportunity to push the importance of their own area of computing. One could hardly have expected anything else. The result was the kind of global behaviour that often emerges from interactions of agents who are seeking to maximize their own local goals. 'Fifth Generation' meant different things to different people. Nearly everyone seemed to have an opinion about what it meant, what key problems it faced, and the best way to address them. Very few seemed actually to have read the published Fifth Generation Project proposals, and indeed regarded them as irrelevant. In his short autobiographical piece, Bob summarises the outcome in these words: "In the end, by the time the Alvey Committee produced its recommendations, virtually every area of Computing and related Electronics was singled out for special promotion."

The UK declined the Japanese invitation to participate in the Fifth Generation Project and set up the Alvey Programme instead. As Bob puts it: "after much more argumentation and discussion, logic programming was identified, along with all the other areas, as worthy of special promotion."

And so, along with many other groups in computing and information technology in the UK, the Logic Programming Group at Imperial College received a large injection of funding under the Alvey Programme—sometimes at the price of forced collaborations that we would not have chosen ourselves—and under the ESPRIT programme of research from the European Commission that followed shortly after. In the mid-1980s the Logic Programming Group had grown to about 50 persons including faculty members, research assistants, PhD students, and support staff. Bob calculates there were 13 separate three-year research grants running at one time, which is my estimate too.

At the time I did not think so much about it, but looking back I stand in awe at the administrative effort that all this required. At the same time, there were new MSc courses being set up in the Department. There were committees, national and international. There were constant demands on Bob's time for invited talks, offers of collaborations, serious and otherwise, letters and articles to respond to (serious and otherwise). There were interviews for newspaper articles. Once, standing in for Bob when he was away, I was interviewed for an article on logic programming and the Fifth Generation for *Vogue* magazine. I declined to unbutton my shirt for the photograph but pouted in the required manner. The industrialist Clive Sinclair was a regular visitor—a version of Frank McCabe's microProlog was eventually released for the Sinclair Spectrum.

There were also difficulties to contend with at the Departmental level. The expansion of the Logic Programming Group, and of some of the other groups in the Department under Alvey and ESPRIT, were causing resentment and some tension. It was perhaps most acute for the Logic Programming Group because we were receiving offers and opportunities to establish ourselves as

an independent entity within the Department, and this was not universally regarded as a healthy development. These matters intruded greatly on Bob's time and energy and caused him much personal stress.

I look through Bob's CV and I am astonished that he found time for any research at all during this period. Yet we had regular technical meetings of various sub-groups one or two times a week. Bob participated actively in projects developing computational logic as a language for school children, on representing laws and regulations, on applications in temporal reasoning, on meta-level reasoning, on abduction, on integrity constraints in databases. How he managed to fit all this in with his other commitments remains a mystery to me (though that will not stop me speculating on it later in this article).

Funding agencies, perhaps only in Europe, like to refer to something called 'critical mass'. Much is made of this, and of its importance when building research activity. Whole research strategies and funding programmes are designed with the goal of creating it. I am not sure where the concept came from, but if it does really exist, I think it must be much, much smaller than is generally assumed. In the case of the Logic Programming Group at Imperial we attained critical mass very quickly. Fission followed shortly after. First we lost contact with the functional programming group—no more time for joint seminars, no more time for conversations in the common room or in corridors. Then the Logic Programming Group divided (harmoniously) into two parts: the Parlog group, working on concurrent Prologs, and the rest, working on everything else. Then the second group split again, this time along no obvious technical boundaries.

In the 1990s, the size of the Logic Programming Group began to dwindle as members of the group moved away to take up positions elsewhere and logic programming became less fashionable. We still had a very sizeable presence in the Department, though it is difficult to count exactly because the boundaries had become very blurred. Notable acquisitions included Dov Gabbay who had arrived in 1983 as a Visiting Fellow and then eventually became a Professor in the Department, and Barry Richards who had moved from the Centre for Cognitive Science at Edinburgh to take up another Professorship. Tensions in the Department abated, or rather, shifted to a different battleground.

From 1989 to 1991 Bob was co-ordinator of the Compulog project, a large collaborative project funded by the European Commission bringing together the main academic groups working in logic programming in Europe. The project was addressing the topics in computational logic closest to Bob's heart. When asked, and sometimes when not asked, I used to say that the technical objectives of the Compulog project were to develop the second half of Bob's *Logic for Problem Solving*. This was a joke (and an exaggeration) but it is true that the Compulog project allowed Bob to extricate himself from Departmental politics and focus his energies on his favourite research topics. The Compulog project funded a replacement for his teaching duties in the Department. A similar arrangement in a project on abductive logic programming funded by Fujitsu continued to provide an academic replacement for another three years. By the time Bob resumed full duties in the Department, in 1994 or so, his rehabilitation, as he

puts it, was complete.

In March 1997 Bob was persuaded to take on the role of Head of the Department of Computing at Imperial College. The Head of Department is essentially a managerial and administrative position, usually for a fixed term, which the Head can organise according to his or her own tastes. It has wide-ranging power and authority but also huge responsibilities for the running of virtually every element of the Department. We were at the time in a period of unrest following the resignation of the previous Head. Bob had gone to speak to the Rector about how the Headship could be resolved, and came back from that meeting finding that he had agreed to take on the job himself. I believe I was the first person he spoke to on his return to the Department. I am not sure which of us was more surprised at the news. The agreement was that Bob's was to be an interim appointment, for three years or so. The Rector's calculation was that Bob's seniority and academic reputation would command authority and respect within the Department. This was a good idea. Bob's calculation was that the time taken away from research for administration and management would be compensated by a reduction in time spent teaching. This was a very good idea in theory. He also thought that it might afford a chance to develop his technical interests, in that it provided an opportunity to test out how ideas from computational logic could serve as a tool in organising the affairs of the Department and in the resolution of conflicts and disputes. This was not such a good idea, even in theory, in my opinion.

Bob threw himself into his new role with typical energy and vigour. The atmosphere in the Department improved considerably. But the day-to-day running of the Department, and a series of obstacles to getting things organised as he wanted, were leaving Bob increasingly frustrated. The theory that time spent on administration and management could still leave time for research was being refuted every day. Eventually, Bob asked to step down as Head of Department after two years not three, and asked to take early retirement. From 1st September 1999 he has been a Senior Research Fellow in the Department of Computing and Emeritus Professor. He has an office in the Department and continues to participate in research projects but has no other duties or responsibilities imposed upon him beyond those he chooses to take on voluntarily. To my eyes, he has attained a kind of blissful state of existence which even his Jesuit teachers might have difficulty claiming could exist.

At some time in the 1980s Bob acquired a small cottage near Petworth in Sussex, which lies in the countryside roughly half-way between London and the South Coast of England. It was a base for weekend breaks and walks in the South Downs. There are several logic programmers around the world for whom that cottage was home during visits spent at Imperial College. Over the years the cottage in Petworth has been extended and developed. Since Bob's retirement, it has been extended again and has now become Bob and Danusia's main residence. Between taking up invitations for extended visits to research

institutes abroad Bob spends his time in Petworth with occasional visits to the Department. He is working on a new book.

Research themes

Bob's early work was in automated theorem proving, where he made contributions to the technology of resolution theorem proving. His connection graph proof procedure[9] provided a general and very efficient framework for reasoning with (full) clausal form theories. By picking links in the graph in different ways, a wide range of reasoning strategies could be accommodated, for non-Horn as well as Horn clause reasoning. These are demonstrated in *Logic for Problem Solving*.

However, it is the special case of SL-resolution which came to dominate later, of course, and which led to the logic programming model of computation. It should be remembered that the extended case for logic programming as a new foundation for computing was developed not by appeal to novel and exotic applications in knowledge processing but by showing carefully how cleanly and elegantly it dealt with standard computing problems and algorithms. The beauty of Bob's *Algorithm = Logic + Control* lies in the detailed exposition of how both *Logic* and *Control* components can be varied to generate families of algorithms.

However, it has always been Bob's contention—passion—that computational forms of logic have much wider application than to the solution of mere computing problems. The single strongest and most sustained driving force in his research has been the goal of developing appropriate forms of logic to make it an effective tool for improving human affairs and communication, and to present these forms in a way that makes them accessible to the widest possible group. These aims reflect his lifelong interests in problem solving and communication, in epistemology and in the philosophy of science. These elements were already evident in the second part of *Logic for Problem Solving* which addresses knowledge representation, problem solving strategies, temporal reasoning and planning, knowledge assimilation and belief revision. His working hypothesis is that the features which make special forms of logic suitable for computational purposes are also the features that will be most natural and effective for use in human problem solving and communication. Application and testing and refinement of this hypothesis is the recurrent theme in his research.

One clear example of these general aims is the sustained project Bob conducted on developing simplified forms of logic and logic programming for school children[10]. In 1978 Bob started a course of logic lessons for 12 year old chil-

[9] Kowalski, R.A. A proof procedure using connection graphs. *JACM* 23(4):733–742 (1976).

[10] Kowalski, R.A. Logic as a Computer Language for Children. In *Proc. European Conference on Artificial Intelligence*, Orsay, France, July 1982. Reprinted in *New Horizons in Educational Computing* (M. Yazdani, ed), Ellis Horwood Ltd, Chichester, 1984, pp121–144. Reprinted in *Progress in Artificial Intelligence* (L. Steels, J.A. Campbell, eds), Ellis Horwood Ltd, Chichester.

dren at his daughters' school. Logic problems were formulated and then solved using Prolog over a telephone connection to a computer at Imperial College. The project was subsequently maintained for about 5 years from 1980 by grants from the Science Research Council and then the Nuffield Foundation and Sinclair Research. The first phase supported Frank McCabe's developments of his microProlog system for micro-processors and the associated programming and query environment ('SIMPLE'). Richard Ennals conducted the lessons and prepared teaching materials for pupils and teachers. If I recall rightly, there were two groups of children, 8 year olds and 12 year olds, and a smaller group of 17–18 year olds. The aim was not just to teach logic as a programming language, but rather to engage the children in developing its use as a representational and reasoning tool in subjects across the whole curriculum. Richard Ennals's own specialty, for example, was History. I am not in a position to comment on the long term impact of the school lessons on the children. It would be interesting to track them down and ask them now what they thought of those lessons. What is clear is that the schools project was instrumental in driving the developments of microProlog and its associated software environments, and in practical knowledge representation techniques that were subsequently used in a variety of other applications.

One such group of applications was in the representation of laws and regulations. I find myself about to write much more about this topic than the others, but this is because it provides the clearest example of Bob's ideas about the applications of logic programming to the world outside computing, and the clearest example of how his stance has been misinterpreted by some of his critics.

In 1979 Bob was invited to participate in a workshop on Computers and Law held in Swansea, in Wales. Although he could not attend, that invitation led to a number of very valuable contacts in the AI and Law community. It soon became clear to us that logic programming provided a general solution to some problems of representation that were being attacked by low-level programming languages or special-purpose formalisms. Our argument was that logic programming provided a better foundation for such developments. We were able to show, for example, how large and complex bodies of definitional law ('qualification norms') can be represented and executed as logic programs. Our representation of the British Nationality Act 1981 is the best known and most commonly cited example[11]. It was originally suggested by Chris Moss, a member of our group, who had been given a draft copy of the Bill while it was still at an early stage of discussion by Parliament. The Bill was very controversial at the time. It proposed to introduce four new categories of British citizenship to replace the existing definition completely, and had been accused by several political groups of being racist in that it disadvantaged certain groups of potential citizens but not others. One of these pressure groups had suggested to us that a formal representation might help to bring this out. We knew that it could not, since whether the Act was racist or not depended on background

[11] Sergot, M.J., Sadri, F., Kowalski, R.A., Kriwaczek, F.R., Hammond, P., Cory, T. The British Nationality Act as a Logic Program. *CACM* 29(5):370–386 (1986).

information about the various categories of persons affected, and that information was not part of the legislation itself. We did subsequently explore, in a different project, whether given the necessary background information, we could predict some of the socio-economic consequences of introducing new legislation, but that was later and was never attempted for the British Nationality Act. However, the British Nationality Act was very suitable for other reasons. It was almost entirely definitional, that is to say, its main purpose was to set out definitions of new legal categories and relationships, which made it amenable to representation as a logic program, yet it was complicated and big so one could see what would be gained from translating it into an executable form. We had already constructed a small demonstration system dealing with the core definitions from Chris Moss's copy of the draft Bill. Frank McCabe, as I recall, was particularly keen that we should continue to develop a larger system dealing with the whole Act to demonstrate that a sizeable application could be implemented using these techniques and his microProlog system. Fariba Sadri, who was about to start a PhD in our group, was employed on funds left over from some other grant to extend the prototype to a more complete representation over two or three months in the summer before she started her PhD. The whole system, including the APES software used to execute the representation, ran on a small micro-computer with only 64K of memory. I used to say that for us at Imperial College, Fifth Generation computing meant any computer with more than 64K of memory.

The work on the British Nationality Act was generally well received and well regarded by the research community in Artificial Intelligence and Law, which shared the pre-suppositions and starting assumptions, and by the lawyers and government agencies with whom we produced various other applications. It did attract negative publicity as well. In the climate of Alvey and the Fifth Generation there was even an article in *The Guardian* national newspaper about it. It repeated a common criticism, that by attempting to represent legal rules as executable logic clauses we were, deliberately or out of ignorance, oversimplifying and mistakenly thinking we could reduce legal decision making to the mechanical application of fixed rules. We were accused of demonstrating a complete ignorance of legal theory and jurisprudence, and a fundamental misunderstanding of the nature of legal reasoning and the process of law. We thought that in describing the work we had identified the background assumptions, and also the limitations of what we had described, but these qualifications had obviously not registered with some critics. That was tiresome enough, but the article went on—to accuse us of being apologists for the racist policies of a right-wing government, and of grabbing government funding for these activities, out of greed or naïvety or both. It even raised the spectre of computers at Heathrow Airport that would decide who would be admitted into the UK and who would not. Even allowing for journalistic licence, these claims were so outrageous (and so completely wrong on every point of fact) that we felt obliged to write a letter of complaint to *The Guardian* in our own defence. I say 'we' though I am not sure now whether Bob wrote on his own or whether it was a joint reply. Perhaps we sent more than one letter. A short flurry of further correspondence ensued.

Bob has used the representation of legislation and regulations as a rich source of motivating examples for developments in the treatment of general rules and exceptions in logic programs[12], and later in his work on the theory of argumentation[13]. He has also been enthusiastic about using examples from legislation to support his views about the value of logic in clarifying and communicating statements of rules in natural language, whether these rules are intended for execution in a computer program or not[14]. It is presumably these general views that have irritated his critics.

For my own part, I learned long ago to avoid making reference to 'AI and law' or to 'logic and law' when asked in casual conversations, at parties and so on, what I am working on. A mention of 'Artificial Intelligence' is often bad enough, but 'Artificial Intelligence and Law' seems to be one of those topics on which everybody has an opinion. Once my car was hit by a Frenchman who drove his car backwards the wrong way out of a one-way street in the area around Imperial College and while we were waiting to sort out the insurance details, he lectured me for half an hour on the futility of AI applied to law. Apparently, I was seriously underestimating the problems. I confess that on that occasion, and others, I have resorted to sarcasm. "Oh no! Ten/fifteen/twenty years I have worked in this area. The law is not just black-and-white? I never noticed. You have opened my eyes. I see now that I have been wasting my time. You are right. I will abandon it." Why any intelligent person should automatically assume that another intelligent person has never noticed that law is not 'black-and-white' and that justice is not dispensed by the mechanical application of fixed rules is the really intriguing question.

It is a facet of Bob's character that he is prepared to take a dose of his own medicine. So for example, at the time he was engaged in Alvey and other grant-awarding committees in the 1980s, he had the idea that the decision making could be improved and made more consistent by formulating clear rules about what projects would or would not qualify for funding. He even formulated a draft set of such rules. He tried essentially the same idea when Head of Department for rationalising teaching and resource allocations. But it is a fundamental misunderstanding of Bob's position to think that such rules are intended to be applied blindly and mechanically. The idea is quite different. One applies the rules to a particular case and examines the conclusion. If the conclusion is unacceptable, or if someone wishes to disagree with the conclusion, the burden is to argue why the rules should not apply in this case. If someone wishes to argue that one or other of the conditions should be ignored or altered, the burden is on

[12]Kowalski, R.A., Sadri, F. Logic programming with exceptions. In *Proc. 7th International Conference on Logic Programming* (D.H.D. Warren, P. Szeredi, eds). MIT Press, 1990, pp598–613. Also in *New Generation Computing* 9(3–4):387–400 (1991)

[13]Kowalski, R.A., Toni, F. Abstract argumentation. *Journal of Artificial Intelligence and Law* 4:275–296 (1996). Also in *Logical Models of Legal Argumentation* (H. Prakken, G. Sartor, eds). Kluwer Academic Publishers, 1997

[14]Kowalski, R.A. English as a logic programming language. *New Generation Computing* 8(2):91–93 (1990).

Kowalski, R.A. Legislation as logic programs. In *Logic Programming in Action* (G. Comyn, N.E. Fuchs, M.J. Ratcliffe, eds). Springer-Verlag, 1992, pp203–230.

them to argue why it should be so altered in this case. The rules serve as a device for structuring the discussion. They are intended to expose the arguments and open up the decisions to scrutiny. There is more to it than that—one might examine the reasons why such a reasonable suggestion does not usually work in practice or why it almost always meets with strong resistance—but it is not my purpose here to give a complete account. I just wanted to give some indication of why Bob's views on 'clear rules' are not nearly as unsophisticated as some critics have assumed.

A strand of research that attracted less criticism was our joint work on the event calculus[15], an approach to representing the effects of action and change in a logic programming framework. It is another example of something that is intended to straddle knowledge representation in AI and problems in mainstream computing, such as temporal databases and database updates. The name was coined (by Bob) to draw attention to the contrast with the conception of action and change employed in the situation calculus of McCarthy and Hayes. Instead of thinking primarily in terms of situations—states of the world at which nothing changes—and actions as transitions between situations, we wanted to think first and foremost about the occurrences of actions—events—and the periods of time that they initiate and terminate; situations during which nothing changes are incidental and there is usually nothing interesting to say about them. Although not stressed in more recent presentations of the event calculus, most of the effort went into deriving an effective computational framework from a general account of events and periods of time and their properties. As in much of his other work, Bob was particularly keen that the presentation should be made as generally accessible as possible. I remember more than one discussion about how abstract and technical the presentation should be. The event calculus was generally well received—at least there were no articles in *The Guardian* about it. Variations, applications, and large scale implementations were subsequently developed in a number of other projects, including as a main strand of a European Community ESPRIT project on temporal and qualitative reasoning. Bob's main applied work in that project was an application to air traffic flow management.

The formal treatment of action and change, and the associated problems of default reasoning and exception handling, have been a constant throughout Bob's research career. These questions are as prominent in his latest research on multi-agent systems as they were in his early work on knowledge representation. I can still cite 'Chapter 6' of *Logic for Problem Solving* without having to look at the Table of Contents. These are issues that are at the heart of knowledge representation. Opinions about their relative merits will vary, but together with the situation calculus (in its many various forms), the event calculus (in its many various forms) continues to be a major driving force for foundational developments in knowledge representation.

[15]Kowalski, R.A., Sergot, M.J. A logic-based calculus of events. *New Generation Computing* 4(1):67–95 (1986). Reprinted in *Knowledge Base Management Systems* (C. Thanos, J.W. Schmidt, eds). Springer-Verlag, pp23–51.

In 1981 Bob visited Syracuse University for a short, one academic term, sabbatical. Whilst there he collaborated with Ken Bowen on amalgamating object-level and meta-level logic programming. Their joint paper[16] was frequently cited in later years in the context of 'meta-level programming' and 'meta-level interpreters' though it was really about something quite different. The goal was to combine the two levels in such a way that they could interact, so yielding a very general and very expressive representational and reasoning framework. The main technical problem was to achieve this interaction without introducing inconsistencies. The Bowen-Kowalski paper laid out the basic moves. Bob continued the investigations with a PhD student, Kave Eshghi, and worked at the applications, to default and epistemic reasoning in particular, until about the mid-1990s. Meta-level inference was a strand of the Compulog project—the Goedel language of John Lloyd and colleagues is a direct descendant—and was a main theme of Bob's MSc course on knowledge representation in the 1990s. With Kave Eshghi Bob also investigated alternative accounts of negation by failure[17], combining ideas from the amalgamated object-level/meta-level work and from abductive reasoning.

Abductive logic programming became increasingly important in Bob's research in the 1990s. It was embraced partly to support reasoning from effect to possible causes, but also because the abductive proof procedures, when combined with a treatment of integrity constraints, provided a computational system that could overcome limitations of standard logic programming systems. I have noticed over the years that Bob has a strong distaste for classical disjunctive reasoning. It may be that an attraction of abductive logic programming is that it provides an alternative way of dealing with disjunctive reasoning. Collaborations with Francesca Toni and Tony Kakas developed an abstract account of the abductive framework[18], which in turn made connections to results emerging in the theory of argumentation. The key idea here is that an argument, to be admissible, must be able to defend itself against attack from other arguments and itself. Varying the details yields argumentation frameworks with different technical properties. Work with Francesca Toni, Phan Minh Dung, and Andrei Bondarenko produced an argumentation-theoretic account of negation as failure, and then more generally, an abstract argumentation framework which includes many of the schemes for default reasoning as special cases[19].

In recent years Bob's interests have turned to multi-agent systems. Here the

[16]Bowen, K., Kowalski, R.A. Amalgamating language and meta-language in logic programming. In *Logic Programming* (K.L. Clark, S-Å. Tarnlund, eds). Academic Press, 1982, pp153–172.

[17]Eshghi, K., Kowalski, R.A. Abduction compared with negation by failure. In *Proc. 6th International Conference on Logic Programming* (G. Levi, M. Martelli, eds). MIT Press, 1989, pp234–254.

[18]Kakas, T., Kowalski, R.A., Toni, F. Abductive logic programming. *Journal of Logic and Computation* 2(6):719–770 (1992).

[19]Bondarenko, A., Dung, P.M., Kowalski, R.A., Toni, F. An abstract argumentation-theoretic approach to default reasoning. *Journal of Artificial Intelligence* 93(1–2):63–101 (1997).

aim has been to combine pro-active, rational problem solving with the reactive behaviour of an agent situated in a changing environment through which it also interacts with other agents. This brings together several of the recurring themes of Bob's research: goal-directed problem solving, the treatment of actions and procedures, belief revision and the assimilation of new facts, and a search for a way of reconciling and integrating two apparently conflicting models of computation. With Fariba Sadri, Bob has been developing a general account which combines a logic programming model of execution with a condition-action execution cycle[20]. His longer term plans are to investigate systematic methods for conflict resolution in multi-agent systems.

Some personal traits

This portrait would not be complete without some glimpse of Bob's personal characteristics. I make no attempt to identify them all, but three in particular stand out for me. First, there is his dogged determination and self-discipline, and the passion with which he embraces scientific concepts and theories. Second there is his tolerance and sense of fair play, which is also connected to the way he has coped with his celebrity. And third there is the question of his sense of humour.

Bob is the most determined and self-disciplined person I have worked with. He will say, no doubt, that he is not self-disciplined because he has temptations and weaknesses. That is irrelevant. When I looked through his CV in preparation of this article, the list of invited talks and travels alone seemed enough for a full-time occupation. I think what impresses me most in this regard is his discipline in dealing with tedious and time-consuming chores which others might put off or simply fail to discharge conscientiously. Bob seems able to dispatch them all with the minimum of fuss.

I have written papers and grant proposals with many different co-authors and have seen other groups in action. All of them seem to experience the same last-minute frenzy as the deadline approaches (and as the editors of this volume would say, passes and recedes into the distance). Once when in the grip of three converging deadlines, I was moaning to Bob about the strain and complaining that everyone seemed to pick the same times for deadlines. Bob's reaction was to ask why I did not set myself my own deadline one week before each piece was due and thereby avoid the last-minute stresses. Parkinson's law does not apply to Bob.

Bob recounts that when he was a student at the University of Chicago, he obtained A grades in all his subjects, except for English writing skills in which he

[20]Kowalski, R.A., Sadri, F. Towards a unified agent architecture that combines rationality with reactivity. *Proc. International Workshop on Logic in Databases*, San Miniato, Italy. Springer-Verlag LNCS 1154, 1996, pp131–150.

Kowalski, R.A., Sadri, F. From logic programming to multi-agent systems. *Annals of Mathematics and Artificial Intelligence* 25:391–419 (1999).

did badly. Many of us would have shrugged our shoulders and dismissed it—"I wasn't really trying/taking it seriously", "I am no good at it". Bob's response was to set about an analysis of what had gone wrong, to diagnose the sources of the problem and to devise methods for overcoming them. This was no easy fix but something that he worked at over several years, and indeed continues to think about still from time to time. When he was Head of Department, for example, he set up a voluntary writing class for the PhD students. I do not know what he told them exactly, but it must have been interesting, for eighteen months after his retirement we still see PhD students searching plaintively for the writing class. At the annual meeting at which we ask the PhD students how their lives could be improved, the most common request was for a resumption of the writing classes by Professor Kowalski.

This same determination and single-mindedness is evident also throughout Bob's technical work. His ability to take up an idea and then to apply it and refine it and pursue it relentlessly is a major strength. Which is not to say that he is always right, or refuses to change his views as a matter of principle. As in the case of writing skills, when ideas do not get the same A grades as others, they are subjected to thorough scrutiny and diagnosis and careful correction.

The passion and conviction with which Bob expounds his technical position can be misinterpreted. In invited talks especially, he will sometimes deliberately adopt an extreme point of view in order to provoke debate or to rehearse the arguments that can be put forward for it. This has apparently led some to assume that his views must be based on some kind of irrational emotional attachment, and that with it must come a refusal to acknowledge the worth of alternative points of view. Nothing could be further from the truth.

Bob is a widely recognised figure in computer science. His name appears, deservedly, in most summaries of accomplishments and trends in computer science, and in logic. This is why he receives requests from students asking for biographical details they need for their project assignments.

The other side of celebrity, however, is that it attracts criticism and caricature. For example, one article, in a 1987 volume of collected papers on the sociology of research in AI and what it called the 'AI establishment', went so far as to compare Bob with a now obscure 16th century figure, Petrus Ramus[21]. Ramus, according to the article, devised distorted and simplified forms of logic or 'method' which he and his followers vigorously promoted for use across all scholarly disciplines. The Ramist method, now all but forgotten (except perhaps in the sociology of science where references to it seem to be quite common), had a very widespread influence for a considerable time across the post-medieval world. It is generally regarded as a curiosity and something of an aberration in the history of logic and rhetoric, which I suppose is the point of the caricature. So in that article parallels are seen between Bob and the figure of Ramus himself, in the 'close technical analogy with the methods of Ramus and Kowalski', in

[21] Philip Leith. Involvement, Detachment and Programming: The Belief in Prolog. In *The Question of Artificial Intelligence*, (Brian Bloomfield, ed), Croom Helm, London 1987.

their widespread influences, particularly over 'impatient and not too profound thinkers', and in the lack of scientific detachment in the disciples of Ramus on the one hand and the esoteric circle of Kowalski's followers on the other hand. The Logic Programming Group at Imperial College is described in these terms:

> Within the academic software teaching and research group it seems—to the outsider—that the entire department is involved in logic programming. Some are involved in the theoretical issues (Clark and Hogger, for example) and some are involved in more practical issues (Ennals, Sergot and Hammond). Kowalski, to some extent, appears to stand above the details of logic programming, leaving the particulars to the group. His role is that of advocate for logic programming, a role which he plays out through academic and commercial contacts and consultancies and through involvement in the provision of research funds as a member of an Alvey advisory committee. It would seem to be difficult for any member of that group to move away from such a logic programming hegemony, for a scientific establishment based upon that logic programming technique must be expected to control its members.

There is nothing in the picture painted here that I recognise. I have no idea where the author got the idea of a hegemony, or what made him think that members were subject to some kind of control. The other facts quoted with such authority are wrong too. Why did the author not bother to check them? The general nature of the remarks in that article, and the repeated references to funding agencies and Bob's influence over the distribution of research funds, leads me to think that the objectives of the article were not entirely scientific.

It is ironic that amongst his most vehement critics are persons whom Bob has defended and supported, usually without their knowledge. And in contrast to the picture painted above, Bob is no seeker of self-publicity. He is very sensitive that collaborators and co-authors should receive their share of recognition for joint work. When the Association for Logic Programming was formed in 1986 it was typical that Bob preferred to take the role of Secretary rather than that of President.

Indeed, if I had any criticism of Bob in this regard, it would be that his sense of fair play can be too acute, and has been taken advantage of. When he was Head of Department, for example, he would never, as a matter of principle, push through by force what he could not obtain by reasoned argument. On occasion, when forming committees or taking advice, he deliberately under-represented his own position and strengthened the representation of opposing views in an effort to give the fairest possible hearing to all. Unfortunately, not everyone is as scrupulous.

I turn finally to the question of Bob's sense of humour. Some of my colleagues will say that making remarks about this is like commenting on the appearance of the current King of France. That is an over-simplification. Bob enjoys jokes

very much, but never tells them. He prefers something that might be called the meta-joke.

For example, I remember when Bob was asked to be the Banquet Speaker at the Conference on Automated Deduction (CADE) in Oxford in 1986. Bob had agreed but was far from happy about it. He dislikes this kind of speaking and finds it very awkward. I am not sure why. When he was Head of Department he was often called upon to make little speeches and introductions, and always found a way of doing them with elegance and wit. For the CADE speech Bob asked my advice, or rather, he wanted suggestions for jokes he could include in his speech, ideally but not necessarily something connected with deduction or reasoning. "Don't worry about it", I said. "It's like a wedding. Everyone wants to be amused. Most of them will be half-drunk. Whatever you say they will laugh. The contents don't matter." I suggested a couple of jokes he could use, with the best reserved for the opening and the end. "You also need a packer", I said. "Something to keep things going in the middle. It doesn't have to be very funny. At that point they will all be laughing anyway, and you just need something to keep things moving along. By the time they realise it isn't funny, you will be into your closing part and they won't notice." Bob looked dubious. "Trust me", I said.

I remembered a (not very funny) joke Dov Gabbay had told me about a young man who wants to become the student of a famous rabbinical scholar, an expert in the interpretation of Talmudic texts. The young man goes along and asks if he may be allowed to study at the Master's feet. "Perhaps", says the Master, "but first you must pass a test." The student agrees. "Two men climb down a chimney", says the Master. "One comes out dirty, the other comes out clean. Which one washes?" "That's easy", says the student. "The dirty one." "No", says the Master. "The clean one. For consider: the dirty one will look at the clean one and will think 'If he is clean, I must be clean.' While the clean one will look at the dirty one and will think 'If he is dirty, I must dirty.' So the clean one washes." "Give me another chance", says the student. "Very well", says the Master. "Two men climb down a chimney. One comes out dirty, the other comes out clean. Which one washes?" "I know this", says the student. "It is the clean one who washes." "No", says the Master. "It is the dirty one who washes. For consider: the clean one will look at himself and see that he is clean. While the dirty one will look at himself and see that he is dirty. So the dirty one will wash." "Oh no!" says the student. "But please, give me one more chance." "Very well", says the Master. "Two men climb down a chimney. One comes out dirty, the other comes out clean. Which one washes?" "Ah, I think I have it now", says the student. "The dirty one washes." "No, no", says the Master. "I don't think you are cut out for this line of work. How can two men climb down the same chimney, and one come out dirty, the other come out clean?" This is not much of joke, though it was funny when Dov told it, and it is about reasoning, of a sort. Bob was not convinced. "Trust me", I said. "It is a good packer. It will keep them laughing until you get on to the better stuff. Perhaps you can even work in some remark about legal reasoning, or something like that."

The following Monday Bob was back in the office. "How did your Banquet speech go?" I asked. "Disaster!" said Bob. "No-one laughed. Especially not at that joke about the student and the chimney." I was surprised. "It isn't much of a joke, I admit. But it should have been enough to keep them happy for a while." "Of course", said Bob, "I did simplify it a bit. It seemed to me that it contained a lot of redundancy, so I cut it down." According to Bob, he eliminated the redundancy and moved straight to the line "How can two men climb down the same chimney and one come out dirty, the other come out clean?"

I have told this story to many people who know Bob well. They chortle with delight when I get to the part "Of course, I simplified it a bit. There was a lot of redundancy." This is exactly what Bob would say, which is why I have included it in this piece. But what is the real joke here? Fifteen years after that speech, I do not know what Bob said at that banquet in Oxford. I know he was teasing me with the reference to redundancy, but I do not know whether he made the same remark in his speech, or whether he mentioned the student and the chimney at all. It is a meta-joke, at my expense.

Conclusion

As part of my subtle interrogations for this article, I asked Bob if he could summarise the various phases of his professional career by picking out an event or anecdote that he would associate with each period of time. "What springs to mind when I mention, say, Edinburgh in the 1970s?", I asked. Bob's answers were as follows: Edinburgh in the 1970s—foundations of logic programming; Imperial College in the 1970s—building up the group and finishing the book; 1980s—the Fifth Generation Project and the Alvey Programme; 1990s—realisation that the early promise of logic programming was not going to be fulfilled, disappointment, and retrenchment (my word); the first years of the 21st century—waiting.

Now I could not let this pass without comment. I understand what Bob means when he says 'disappointment'. He is referring to the prospects of logic programming as the unifying foundation for all of computing, and to the influence of computational logic on the world outside computing. But honestly I cannot see anything to be disappointed about.

In recent years Bob has given talks with titles along the lines of "Logic programming: Where did it all go wrong?" or "Why was logic programming a failure?". Of course I know that he is being deliberately provocative when choosing such titles and that the point of the talk is usually to identify the technical reasons why logic programming as originally conceived does not measure up to all requirements now. Perhaps I caught him on a bad day, but on the occasion I heard him deliver this talk I believe I detected a genuine tone of disappointment. The title, on that occasion at least, was not entirely ironic.

I confess that my reaction was to laugh (inwardly, of course). All I could think of was the image of George Best, a very famous ex-footballer ('soccer

player') in the UK, and the story he tells about himself on TV chat shows and the like. I hope English readers will forgive me for digging up such a tired old chestnut. I know it is corny but honestly it was the vision that flashed before my eyes at this talk of disappointment. George Best played in the 1960s and early 1970s. He is still internationally regarded as one of the two or three best footballers of all time. His career ended tragically early (tragically for us, not necessarily for him) in alcohol, and nightclubs, and even a short prison sentence. He finished playing when he should have been approaching his peak.

George Best tells the following story about himself. Some years after he had finished playing he was staying at a casino somewhere, in Las Vegas I think, though the details do not matter. He was at that time accompanied by a Miss World, or a former Miss World, or at least a Miss World finalist. I cannot remember. And one evening at this casino he won a considerable sum of money, of the order of $20,000. Again the details do not matter. Back at his hotel suite, while the former Miss World went into the adjoining bathroom, Best spread his winnings, all $20,000 of it, over the bed and phoned room service for champagne. The champagne was delivered by an old Irish waiter who of course recognised George Best immediately. According to Best's story, the waiter looked around the bedroom—the vintage champagne in the ice bucket, the former Miss World emerging from the bathroom, the cash spread all over the bed—and shook his head sadly. "Mr Best," he said, "where did it all go wrong?"

It seems to me that a field which annually has at least one, sometimes two, international scientific conferences devoted to it is not a moribund field. And this is not to count the journals, and the numerous series of workshops and meetings (CLP, LOPSTR, ILP, LPNMR, among others) devoted to specific aspects of logic programming and its applications. While logic programming may not have come to be the foundation for all of computing, that is partly because the conception of computing itself has changed. It is the cornerstone of many important sub-areas of computing, and its influences continue to be felt across all of computer science and AI.

I look at the chapters of this volume spread proverbially across the bed. I think of the many others who would have jumped at the chance to contribute a chapter to this volume. They are the former Miss Worlds peeking around the bathroom door, so to speak. Looking at this I do not shake my head sadly and ask "Where did it all go wrong, Bob?". A better question would be "Where did it all go right, Bob?", except that we know the answer. This volume is a worthy and deserved tribute to someone who has made a lasting contribution to the development of computer science, and ideas far beyond.

An influential coach and all-round good guy. Yes indeed, among many other things.