COMPUTING SCIENCE

Newcastle's French Connection

Brian Randell

TECHNICAL REPORT SERIES

No. CS-TR-1200 **April 2010**

TECHNICAL REPORT SERIES

No. CS-TR-1200 April, 2010

Newcastle's French Connection

Brian Randell

Abstract

Reminiscences of many years of research co-operation between Newcastle University and LAAS-CNRS (Toulouse, France), in particular on dependability concepts and definitions. (Invited presentation at the workshop "Dependability of Computing Systems: Memories and Future" held at LAAS on 15-16 April 2010 to mark the award of the 2009 Grand Prize in Informatics of the French Academy of Sciences to Jean-Claude Laprie.)

Printed and published by the University of Newcastle upon Tyne,
Computing Science, Claremont Tower, Claremont Road,
Newcastle upon Tyne, NE1 7RU, England.

Bibliographical details

RANDELL, B.

Newcastle's French Connection
[By] B Randell
Newcastle upon Tyne: University of Newcastle upon Tyne: Computing Science, 2010.

(University of Newcastle upon Tyne, Computing Science, Technical Report Series, No. CS-TR-1200)

Added entries

UNIVERSITY OF NEWCASTLE UPON TYNE
Computing Science. Technical Report Series. CS-TR-1200

Abstract

Reminiscences of many years of research co-operation between Newcastle University and LAAS-CNRS (Toulouse, France), in particular on dependability concepts and definitions. (Invited presentation at the workshop "Dependability of Computing Systems: Memories and Future" held at LAAS on 15-16 April 2010 to mark the award of the 2009 Grand Prize in Informatics of the French Academy of Sciences to Jean-Claude Laprie.)

About the author

Brian Randell graduated in Mathematics from Imperial College, London in 1957 and joined the English Electric Company where he led a team that implemented a number of compilers, including the Whetstone KDF9 Algol compiler. From 1964 to 1969 he was with IBM in the United States, mainly at the IBM T.J. Watson Research Center, working on operating systems, the design of ultra-high speed computers and computing system design methodology. He then became Professor of Computing Science at the University of Newcastle upon Tyne, where in 1971 he set up the project that initiated research into the possibility of software fault tolerance, and introduced the "recovery block" concept. Subsequent major developments included the Newcastle Connection, and the prototype Distributed Secure System. He has been Principal Investigator on a succession of research projects in reliability and security funded by the Science Research Council (now Engineering and Physical Sciences Research Council), the Ministry of Defence, and the European Strategic Programme of Research in Information Technology (ESPRIT), and now the European Information Society Technologies (IST) Programme. Most recently he has had the role of Project Director of CaberNet (the IST Network of Excellence on Distributed Computing Systems Architectures), and of two IST Research Projects, MAFTIA (Malicious- and Accidental-Fault Tolerance for Internet Applications) and DSoS (Dependable Systems of Systems). He has published nearly two hundred technical papers and reports, and is co-author or editor of seven books. He is now Emeritus Professor of Computing Science, and Senior Research Investigator, at the University of Newcastle upon Tyne.

Keywords

DEPENDABILITY,
CONCEPTS,
LAAS-CNRS,
JEAN-CLAUDE LAPRIE

Newcastle's French Connection

Thursday 15-16 April 2010,

LAAS, Toulouse

Brian Randell <brian.randell@ncl.ac.uk>

Newcastle University

It is a great pleasure, and privilege, to speak at this event in honour of Jean-Claude Laprie, celebrating both his transition to emeritus status, and his winning of the 2009 Grand Prize in Informatics of the French Academy of Sciences.

I have found myself giving several invited talks in recent years in which I've indulged in personal reminiscences, including a lecture here in Toulouse in February 2008 at the Fourth European Congress on Embedded Real Time Software (ERTS), when I was asked to talk abut the initial 1968 NATO Software Engineering Conference. I have mixed feelings about such invitations, since – although I am happy to be classified as a computer historian, albeit a part time one, I am not so pleased by being classified as an historical artefact! However, on an occasion like this it seems entirely appropriate for me to look back and to talk about my numerous very fruitful contacts, over many years, with Jean-Claude Laprie – hence the title of this talk, though the allusion I have in mind is to the "Newcastle Connection" (the name of software we developed years ago for constructing distributed Unix systems), rather than to a certain Oscar-winning film.

In the Beginning

Wearing my computer historian hat, I am all too aware of the frailty of human memory, my own included. It was nevertheless a surprise to me when, quite recently, Jean-Claude corrected (in the sense of falsified) the very clear memory that I had of our first meeting. I had been sure it had occurred when he and a colleague from LAAS came to Newcastle and gave a talk in the mid-1970s about the various large collaborative industry-oriented projects that LAAS was involved in. I recall that this talk seemed a little strange to us, because this was before we had become involved in any large industry-related collaborative research projects, either UK or EU-funded. However it turns out that Jean-Claude was not on this visit, and that instead we first met at the 1979 Euro-IFIP Working Conference on Reliable Computing and Fault Tolerance in London, when there were the discussions, led by Al Avizienis, that led to the creation of IFIP WG 10.4.

I had met Al a few years earlier, at the 1975 International Conference on Reliable Software in Los Angeles, when I presented what was almost the first paper on recovery blocks [Randell 1975]. (The first was at an INRIA conference in Paris the previous year, and very appropriately was co-authored with Mike Melliar-Smith, Jim Horning and Hugh Lauer, since their role in the invention of the recovery block concept was at least as great as mine [Horning, Laur et al 1974].) Al was in fact in the same session as me at the Los

Angeles Conference, and presented a paper on hardware fault tolerance and intolerance – a lovely choice of title [Avizienis 1975].

The other memory I have of that conference was of Harlan Mills, of IBM's Federal Systems Division, giving a talk in which he said that professional programmers should aim, and if well-chosen and trained should be expected always, to produce completely correct software. At the conference's final panel session I remarked that it seemed to me that conference participants divided into two classes – those who accepted, and those who disbelieved, Murphy's Law, "Anything that can go wrong will go wrong". I went on to remark that I had the impression that Harlan Mills had arranged that a local Town Ordinance be passed in Gaithersburg, the home of IBM FSD, repealing Murphy's Law – he had no answer to this!

We at Newcastle went on working on what we learned to call "design fault tolerance", extending our concerns beyond sequential software programs first to parallel then to distributed computing systems. For example, in 1978 the late Phil Merlin and I had a paper, on State Restoration in Distributed Systems [Merlin and Randell 1978], at FTCS-8 in Toulouse, where Al had presented his work on N-Version Programming [Chen and Avizienis 1978]. (I think Phil rather than I attended FTCS-8, so I'm not sure when I first visited Toulouse.) Subsequently, needing an actual platform for our work in fault tolerance in distributed systems, we produced the Newcastle Connection software that I alluded to earlier [Brownbridge, Marshall et al. 1982].

IFIP WG 10.4

By 1979 the idea that fault tolerance was of potential relevance to software as well as hardware was becoming known, if not widely accepted, and I and my Newcastle colleagues were finding the fault tolerance community increasingly relevant and congenial, an adjective that applies particularly to Jean-Claude.

The idea that Al and Jean-Claude promoted at the 1979 Euro-IFIP Conference of a new IFIP Working Group was thus very attractive to me – I think my main contribution to its planning was to urge the relevance of my experience as a founder member of IFIP Working Group 2.3 (on Programming Methodology). WG 2.3 in fact had its origins in the NATO 1968 Software Engineering Conference, and the 1968 resignations over Algol 68 from the IFIP Working Group 2.1 on Algol, two events in which I was very much involved. (1968 was a very interesting year for me!). However, most members of WG 2.3 equated software reliability with program correctness, so by 1979 I was beginning to feel somewhat out of step with my colleagues on this working group.

When we had set up WG 2.3 we had insisted, in the light of our experience in WG 2.1, that the new Working Group retain control over appointments to membership, so as to avoid having to accept political appointees, and we had a rule that potential new members should first be invited to participate at meetings several times as guests, before they could be considered for membership. These strategies were adopted by WG 10.4, where they proved as successful as they had been for WG 2.3.

My ever-inadequate memory had told me that Jean-Claude chaired WG 10.4 from the outset. In fact a little research for this talk soon told me that Al was the first Chairman, and that Jean-Claude did not take over until 1986. Under his influence, the Group has had

many delightful meetings in France. This is something I know he once boasted about to a French Government Minister (taking care to omit certain geographical details). And so, thanks to Jean-Claude, I and a number of my colleagues have become embarrassingly familiar with Martinique and Guadeloupe in particular.

Fault Tolerance Terms and Concepts

WG 10.4 provided very pleasant, as well as very effective, opportunities for intensive discussions on many topics, and in particular on identifying and defining an appropriate set of fundamental concepts and terminology. This was an issue that had exercised us from very early on at Newcastle. Little did I guess how much effort would be expended, including by me, over the years to come in refining and extending the set of dependability terms and concepts. Jean-Claude was *the* major player in much of this work, and has provided detailed accounts of how it developed and was documented, so I feel justified in focussing briefly on just a few issues.

Right from the start of our work at Newcastle on software fault tolerance we had realised the inadequacy, for software, of the fault tolerance terminology used at that time by hardware designers. The problem was that the hardware engineers took as their starting point a set of explicit definitions of a few particular forms of well-known fault, such as "stuck-at-zero" faults, "bridging" faults, etc. We saw no prospect of producing a useful such set for software, so this approach did not seem at all appropriate for thinking about residual software design faults, given the huge variety, and the lack of any useful classification, of such faults.

I well remember Mike Melliar-Smith visiting our Department of Philosophy in search of help, and coming back saying: "they seem to know a lot about truth, but very little about error". In fact, we eventually realised that we our definitional problems could be eased by starting not from faults and errors, but from the concept of a system "failure", from which we could work back to the concepts of faults and errors [Melliar-Smith and Randell 1977].

Since a system might fail in all sorts of ways, e.g., producing wrong results, producing no results at all, running too slowly, revealing confidential information, putting people's lives at risk, etc., we came to regard reliability as a very general topic. Indeed, we argued that safety, security, etc., should all be regarded as mere special cases of reliability. This did not go down at all well with those working in these other domains. It was Jean-Claude, who came to our linguistic rescue by proposing the use of the term "dependability" for the concept underlying our broadened definition of reliability.

Jean-Claude's ability to rescue us linguistically is an indication of his impressive mastery of my mother tongue (which, to my regret, is English rather than Welsh). However Jean-Claude's English though masterly, is not quite impeccable (or *impeccable*), since the workshop programme that he originally provided to me, and indeed the final version we all have today, employs an adjective to describe tonight's reception, namely "dinatory". This is evidently constructed from the French "dinatoire", an adjective that has yet to find its way into any English (or American) dictionary. I nevertheless am utterly confident that tonight's dinatory reception will be highly dependable.

Another contrast in the 1970s to the then-standard hardware fault tolerance definitions was that we found it necessary in our 1977 paper to admit that identifying software errors and faults often involved subjective judgement. (There are not infrequently several different, but equally plausible, ways a given program bug could be repaired – implying several different identifications of the errors and faults.) However the 1977 account did not make a similar comment about the problem of identifying failures. Rather it merely said: "[A system's] service must be specified (but not necessarily pre-specified) . . . a failure of a system occurs when that system does not perform its service in the manner specified, whether because it is unable to perform the service at all, or because the results and the external state are not in accordance with the specifications."

During the early 1980s a joint committee of WG 10.4 and the IEEE Technical Committee on Fault-Tolerant Computing, a committee in which Newcastle and LAAS were both very active, worked intensively on the formulation of an adequate set of concepts and definitions. The 1982 FTCS-12 Conference Proceedings contain a number of papers emanating from this work, in particular the one in which Jean-Claude introduced and motivated the use of the tem "dependability". In this paper [Laprie and Costes 1982] dependability is defined as "the ability of a system to accomplish the tasks (or, equivalently, to provide the service) which are expected of it".

The joint committee's summary paper in FTCS-12 [Lee and Morgan 1982], gets near to the heart of the failure and hence the dependability/reliability definition problem in saying "it may be argued that a disagreement between a system and its specification is a result of an incorrect specification, rather than a problem with the design, implementation or usage of the system. To resolve such conflicts, it is necessary to have some higher authority provide the definitive judgement on the system and its behaviour. We refer to this higher authority as the *Authoritative System Reference* (ASR) for the system. . . A *failure* of a system is an event that occurs when the behaviour of a system deviates from that required by its Authoritative System Reference." I'll return later to the issue of judgement.

Subsequently, over the years various slightly different formulations were adopted for the definition of dependability. For example, in 1986, Al and Jean-Claude [Avizienis and Laprie 1986] used the formulation: "Dependability is that property of a system which allows reliance to be justifiably placed on the service it delivers." Such a definition served well, and was widely-accepted. But I continued to have reservations about I, not just because I felt the system specification issue was not fully resolved, but also because it seemed to me that to employ the term "reliance" in a definition of dependability was rather circular. Of course any dictionary contains many circular definitions – but the circles are normally lengthy enough to take any reader to at least some words they are familiar with already, at which they can usefully stop. (Contrast Knuth's famous pair of index entries "Circular Definition: see Definition, Circular" and "Definition, Circular: see Circular Definition" [Knuth 1968].) I'll return to this circularity issue later.

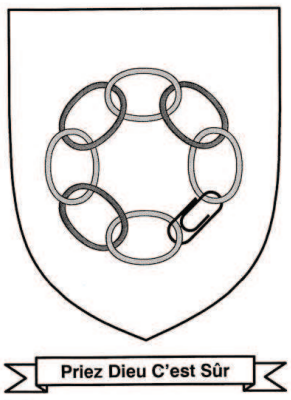
EU Projects

My collaboration with Jean-Claude became much more intense through our co-directorship of a whole series of EU-funded projects, which from time to time provided further opportunities for working together on issues of concepts and definitions. The first

of these projects was PDCS (Predictably Dependable Computing Systems), which started in May 1989. This project had been a long time in gestation – indeed Jean-Claude traces its start to FTCS-17 in Pittsburg in June 1987, when he recalls he and I met at my request with Tom Anderson and Bev Littlewood, and probably also Hermann Kopetz, to discuss a draft project outline that I had prepared.

Prior to and during PDCS's planning period I was very involved in the politicking, with George Metakides, that led to the setting up of the ESPRIT Basic Research Programme, the forerunner of the present IST FET Programme. Eventually however Esprit Basic Research had its first call for proposals, and PDCS was one of the largest to be accepted. The first open workshop was held here at LAAS and was particularly memorable. (This was when I learned that LAAS do things in style – for example, there was a large PDCS flag flying over the LAAS building, and little PDCS flags at each table during the workshop banquet – I still have mine.) These flags used the PDCS logo, created by Jean Arlat, which had won the project logo competition we had judged at the beginning of PDCS during a workshop held in York. (The entry that I created for the competition, a design I was very proud of, lost out to Jean's despite my position as project co-ordinator!)

Jean Arlat's PDCS logo



My spurned design

Jean wasn't present to receive his prize of a large bottle of champagne, though Jean-Claude did agree to take the cork back to Toulouse for him – leading to John McDermid's suggestion that our project's acronym in fact stood for "Prize Drunk, Cork Saved". These sorts of recollections demonstrate that we were putting lots of effort into the social engineering, as well as the computer science, aspects of the project, a strategy that Newcastle and LAAS cooperated on to the full, and to great effect.

One unexpected benefit of PDCS was that it suddenly became evident to me that the problem of when and where I could take my long overdue sabbatical had a very practical and solution. Initially it had looked as though it would have to be further delayed because of my newly-acquired project co-ordination responsibilities. But PDCS caused me to question my assumption that my sabbatical would be most appropriately taken in the USA – a decision that the Director of the ESPRIT Basic Research Programme made much of.

Instead in 1991 I had a very pleasant and very active sabbatical at LAAS, free from various Newcastle responsibilities, and able to concentrate wholly on research and in particular on PDCS. I worked on a number of topics, in particular with Jean-Charles Fabre on extending LAAS's work "Fragmentation-Redundancy-Scattering" [Fabre and Randell 1992]. (I also went on a number of wonderful rando-pédestres – country rambles – with the Pyrénées Club de Toulouse, so ensuring that my growing French vocabulary was not solely devoted to *l'informatique*.)

The PDCS project was very successful, and very well regarded in Brussels. It led to a whole series of follow-on EU-funded projects in which LAAS and Newcastle, and Jean-Claude and I, collaborated. There is no time to discuss all of these projects, so for the moment I will just mention one, namely MAFTIA. (Needless to say, the project name was carefully chosen for the closeness of its acronym to the name of another well-known collaboration, albeit one of Italian origin). The MAFTIA acronym in fact stood for "Malicious- and Accidental-Fault Tolerance for Internet Applications" [Powell, Adelsbach et al. 2001].

What I particularly enjoyed, as we prepared the MAFTIA proposal, was the feeling that once again, just as had been the case when we started work on design fault tolerance, we had identified a potentially important and very interesting, but highly controversial, new challenge for ourselves and the fault tolerance community. My own role in MAFTIA was interrupted by illness, but MAFTIA, whose co-ordination was taken over by my colleague Robert Stroud, was even more successful than PDCS – and saw another burst of activity on concepts and definitions, aimed at making sure they adequately covered the problems of deliberately-induced system faults.

Redefining Dependability

It was only much later, culminating in the intensive collaboration with Jean-Claude and Al, and Carl Landwehr that led to our 2004 paper "Basic Concepts and Taxonomy of Dependable and Secure Computing" [Avizienis 2004], that the essentially-subjective nature of failures and indeed of dependability was made adequately clear, at least to my satisfaction. At the same time we solved what I, at any rate, had long-regarded as the problem of circularity in the typical definitions of dependability. This was by defining failure as, in principle, an event that is *judged* as such by some other system. In other words:

> A system, operating in some particular environment, may fail in the sense that some other system makes or could, in principle, have made a judgement that the activity or inactivity of the given system constitutes *failure*.

And then dependability in the following terms:

> Dependability is the ability to avoid failures that are more frequent and more severe than is acceptable

These definitions are based essentially on just two basic concepts, namely 'system', and 'judgement', for each of which any standard dictionary definition will suffice. The judgemental system is a potential or actual observer of the given system, and may be an automated system, a user or an operator, a relevant judicial authority, or whatever. This

judgemental system may or may not have a detailed system specification to guide it, though evidently one would hope that such a document would be available to guide the system construction task. Ideally such a specification would be complete, consistent, and accurate – in practice there are likely to be many occasions when it is the specification rather than, or as well as, the specified system which is judged as failing. Different judgemental systems might, of course, come to different decisions regarding whether and how the given system has failed, or might fail in the future, in differing circumstances and from different stakeholders' viewpoints. Thus, especially with highly complex systems, the notion of what constitutes a failure and whether one has occurred can often be quite subjective.

Moreover, such a judgemental system might itself fail in the eyes of some other judgemental system, a possibility that is well understood by the legal system, with its hierarchy of courts. So, there is a (recursive) notion of 'failure', which clearly is a relative rather than an absolute notion. So then is the concept of dependability, at any rate when one is dealing with hugely complex systems.

I like to sum up my views on these concepts by saying that systems in principle *always* come in threes – each system has another system as its environment, and is under the surveillance of a further system that can judge whether it is failing. This I find a very pleasing way of looking at things. And of course when one looks *inside* a system . . .

The Resilience Knowledge Base

It had been our perhaps naive hope that the numerous publications that we and others produced describing the fundamental dependability concepts would help to build bridges between the various different research communities that worked on what we regarded as various different aspects of dependability. We had no expectation that people would convert from using their familiar existing terminology. Nevertheless, although the work has had a widespread effect, there is still a distressing tendency for people to invent and define (usually rather casually) new terms for what is essentially dependability.

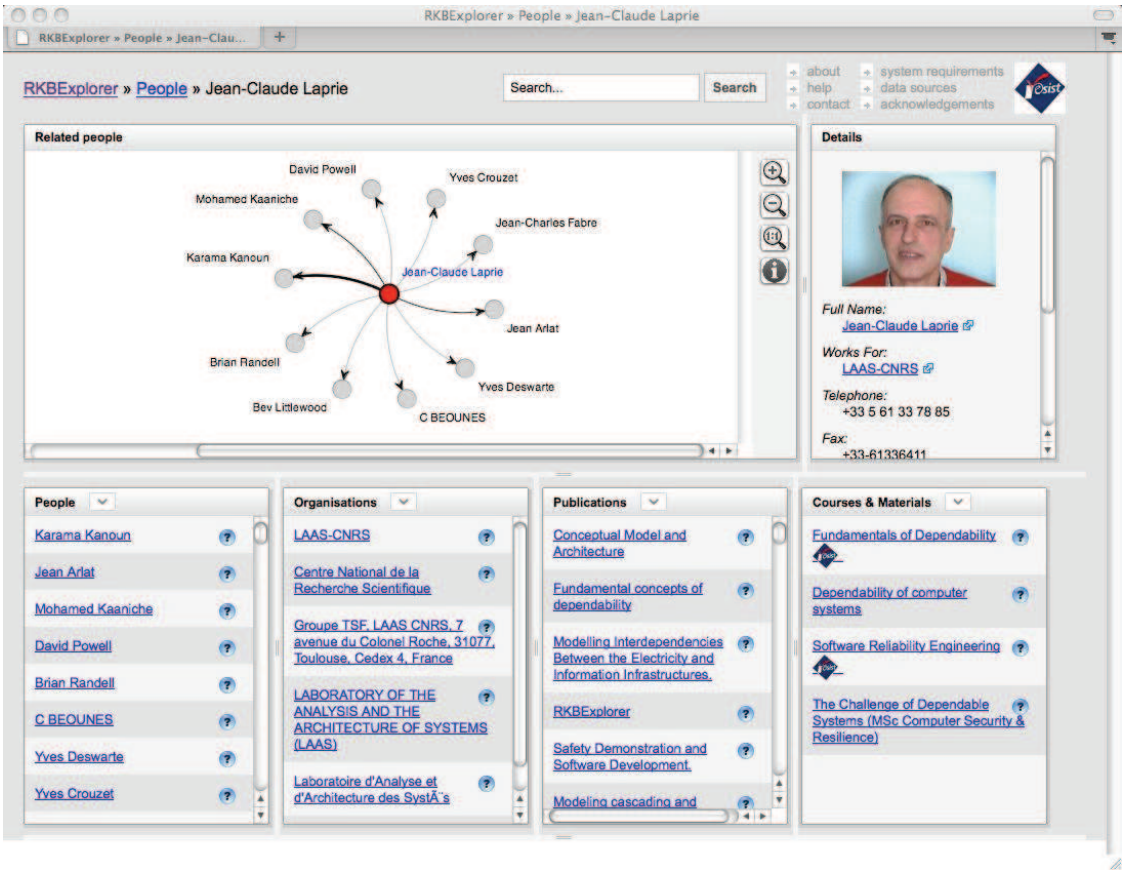
As I have said in various different arenas, there are several reasons, none of them respectable, for introducing and using new terminology when accepted terminology already exists. These reasons include: ignorance; the wish to appear original; the aim of inflating a publication count; the hope of tapping a new or different source of funding, etc. One of the most recent such introductions has been the term "resilience". However, since this came from the IST Directorate in Brussels we were forced to use it – in fact in the most recent EU-funded Newcastle-LAAS Connection, the ReSIST Network of Excellence that Jean-Claude and Karama Kanoun directed. (ReSIST stands for Resilience for Survivability in IST.)

One of ReSIST's chosen activities was to build an integrated knowledge base for the European dependability community, incorporating information about people, publications, projects, institutions, courses, etc. [Glaser, Millard et al. 2008] The building of the RKB was done mainly by Hugh Glaser and Ian Millard of Southampton University, one of ReSIST's 18 partners. This involved Al, Jean-Claude, and me in much reconsideration of terminology, the construction of ontologies, and all sorts of other fun

issues. And it resulted in a very impressive system, though one whose potential has not been adequately exploited.

The RKB pulled together a huge amount of information, over sixty million items in all, from a large number of sources, and dynamically determined what was connected to what. This information came not just from the project partners, but also such major sources as the EU Cordis database, the DBLP Computer Science Bibliography, CiteSeer, the National Science Foundation, the ACM Digital Library, and the RISKS Digest.

The following screenshot of what we termed the RKB Explorer provides a view into the information the RKB had discovered about Jean-Claude, his publications, contacts, etc.

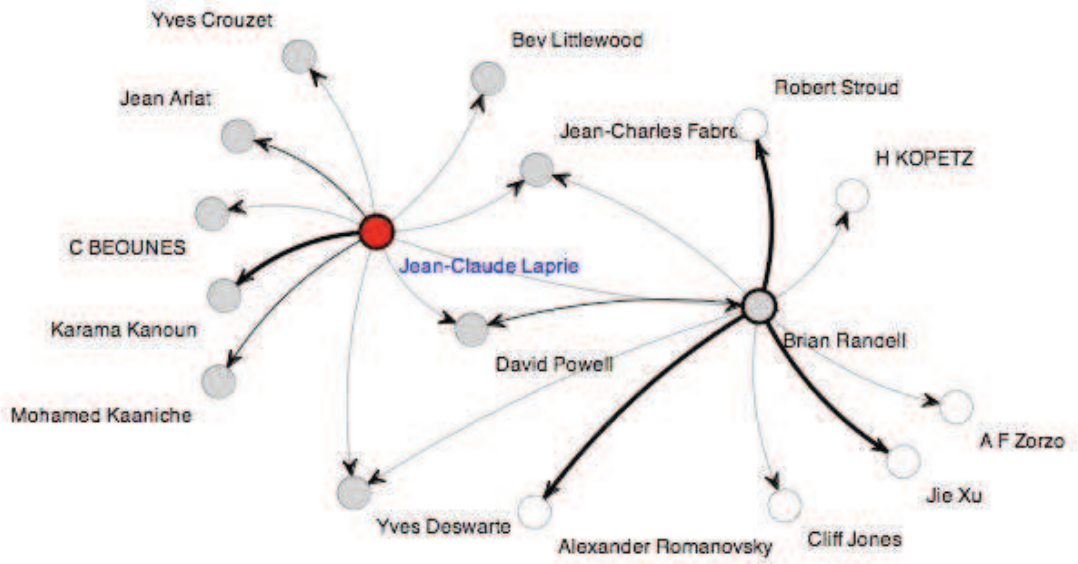


The RKB Explorer Window

The next figure, taken from a pane of another screenshot, portrays the RKB's "knowledge" about how Jean-Claude and I collaborate. The links it shows have been identified as a result of it finding instances of paper co-authorship, common project membership, etc. Similar diagrams can be displayed showing how publications are related to one another, how organisations are related, etc.

One can drill down using these diagrams, or using the lists of items in the columns in the lower half of the Explorer window, and so move around the knowledge base from person to person or to publication or to organisation, etc., and at each point find full details of the links that the RKB has discovered. For example, it will on request provide full details of

17 papers that it had found that Jean-Claude and I had co-authored, and of five projects to which we both had belonged.



Jean-Claude's and my connections, as calculated and portrayed by the RKB

Much effort also went into incorporating into the RKB Explorer facilities for exploring concepts and their relationships, in ways that would transcend terminological variations, something that conventional search engines do not achieve. I had rather optimistic hopes that the RKB would as a result contribute significantly to the proselytizing effects of the dependability community's work on fundamental concepts and definitions. These hopes are, I'm afraid, as yet largely unfulfilled.

Concluding Remarks

This then has been an outline of Newcastle's, in particular my, French Connections – interactions with LAAS and in particular Jean-Claude, over a period of more than thirty years. I'd like to finish by briefly mentioning the work I've been doing most recently with a Newcastle colleague, since this once again, has been motivated and inspired by all the discussions I have had over the years with Jean-Claude.

The starting point was an attempt, using occurrence nets, to formalize the "fault/error/failure" chain, as it could occur among complex evolving systems. The result was the realisation that a further conceptual simplification was possible, since once one considered the event of one system modifying another (e.g. a software developer updating a software module), then it became clear that the distinction between the apparently very different concepts of 'system' and 'state' was essentially just a question of viewpoint. This has led us to develop a theory of "structured occurrence nets", and to plan the development of tools to aid system validation, system synthesis, and failure analysis – but that is a story for another time [Koutny and Randell 2009].

There is a danger that, by having concentrated somewhat on topics related to dependability concepts and definitions, I have given a rather inadequate picture of the extent, variety, and merits of my collaborations with Jean-Claude. For example, I've barely touched on the many ways in which we have shared the joys and tribulations of co-directing various large projects, of coping with Brussels, etc. For this I apologise. Suffice it to say that Jean-Claude has always been the main pillar of Newcastle's French Connection, and that it is a pleasure to acknowledge the debt we all owe to him.

References

[Randell 1975] B. Randell, "System Structure for Software Fault Tolerance," *IEEE Trans. on Software Engineering*, vol. SE-1, no. 2, pp.220-232, 1975.

[Horning, Lauer et al 1974] J.J. Horning, H.C. Lauer et al. "A Program Structure for Error Detection and Recovery," in *Proc. Conf. on Operating Systems, Theoretical and Practical Aspects (Lecture Notes in Computer Science, vol. 16),* pp. 171-187, IRIA, Springer Verlag, 1974.

[Avizienis 1975] A. Avizienis. "Fault-Tolerance and Fault-Intolerance: complementary approaches to reliable computing," in *Proc. Int. Conf. on Reliable Software,* pp. 458-464, Los Angeles, California, 1975.

[Merlin and Randell 1978] P.M. Merlin and B. Randell. "State Restoration in Distributed Systems," in *Proc. 8th Int. Symp. Fault-Tolerant Computing (FTCS-8),* pp. 129-134, Toulouse, IEEE Computer Society Press, 1978.

[Chen and Avizienis 1978] L. Chen and A. Avizienis. "N-Version Programming: A fault-tolerance approach to reliability of software operation," in *Proc. 8th IEEE Int. Symp. on Fault-Tolerant Computing (FTCS-8),* pp. 3-9, Toulouse, France, 1978.

[Brownbridge, Marshall et al. 1982] D.R. Brownbridge, L.F. Marshall et al, "The Newcastle Connection, or - UNIXes of the World Unite!," *Software Practice and Experience*, vol. 12, no. 12, pp.1147-1162, 1982.

[Melliar-Smith and Randell 1977] P.M. Melliar-Smith and B. Randell. "Software Reliability: The role of programmed exception handling," in *Proc. Conf. on Language Design For Reliable Software (ACM SIGPLAN Notices, vol. 12, no. 3, March 1977),* pp. 95-100, Raleigh, ACM, 1977.

[Laprie and Costes 1982] J.C. Laprie and A. Costes. "Dependability: A unifying concept for reliable computing," in *Proc. FTCS-12,* pp. 18-21, Los Angeles, IEEE, 1982.

[Lee and Morgan 1982] P.A. Lee and D. Morgan. "Fundamental Concepts of Fault Tolerant Computing: Progress Report," in *Proc. FTCS-12,* pp. 34-38, Los Angeles, IEEE, 1982.

[Avizienis and Laprie 1986] A. Avizienis and J.C. Laprie, "Dependable Computing: From concepts to design diversity," *Proc. IEEE*, vol. 74, no. 5, pp.629-638, 1986.

[Knuth 1968] D.E. Knuth. The Art of Computer Programming, Vol. 1: Fundamental Algorithms, Reading, MA, Addison-Wesley, 1968, 634 p.

[Fabre and Randell 1992] J.C. Fabre and B. Randell. "An Object-Oriented View of Fragmented Data Processing for Fault and Intrusion Tolerance in Distributed Systems," in *Proc. 2nd European Symp. on Research in Computer Security (ESORICS 92),* pp. 193-208, Toulouse, France, Berlin: Springer-Verlag, 1992.

[Powell, Adelsbach et al. 2001] D. Powell, A. Adelsbach et al. "MAFTIA (Malicious- and Accidental-Fault Tolerance for Internet Applications)," in *Supplement of the 2001* Int. Conf. on Dependable Systems and Networks, pp. D32-D35, Göteborg, Sweden, IEEE Computer Society Press, 2001.

[Avizienis, Laprie et al. 2004] A. Avizienis, J.-C. Laprie et al, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp.11-33, 2004.

[Glaser, Millard et al. 2008] H. Glaser, I. Millard et al. *A Knowledge Base for Dependability and Security Research,* Technical Report 1132, School of Computing Science, Newcastle University, 2008.
http://www.cs.ncl.ac.uk/publications/trs/papers/1132.pdf

[Koutny and Randell 2009] M. Koutny and B. Randell, "Structured Occurrence Nets: A formalism for aiding system failure prevention and analysis techniques," *Fundamenta Informaticae*, vol. 97, no. 1-2, pp.41-91, 2009.