Requirenautics Quarterly

The Newsletter of the Requirements Engineering Specialist Group of British Computer Society

 Issue 15 (October 1998)

RE-Locations

RE-Soundings

Editorial

Rush, rush, rush! If I wasn't in such a hurry to get this issue to you in time to remind you all about CEIRE later this month, then I'd have time to write a proper editorial and tell you about all the wonderful things we have in this issue. Like for example the reviews of the new Kotonya and Sommerville's new book on requirements engineering, and Stevens, Brook, Jackson and Arnold's new book on systems engineering. And the review of the REFSQ workshop. And the article by Ian Alexander on why you need requirements engineering. And of course, another insightful column from Geoff Mullery. And not forgetting the call for papers for RE'99 in Limerick. But as I don't have time, I'll just let you all read on and enjoy them...

Copy deadline for the next issue is 18th December 1998

Steve Easterbrook,
NASA IV&V Facility, Fairmont WV

Chairman's Message

Autumn looks like its here, bringing another bumper issue of RQ! As usual Steve Easterbrook has put together a variety of contributions: books reviews, meeting summaries and regular features. Since the last newsletter, the RESG has organised three events - a UML tutorial, a full-day seminar on Scenarios and Use Cases, and the first joint RESG-IEE colloquium on Managing Requirements Change. after a short summer break, and we're now finalizing details for the follow-up to the hugely successful RE-Day: CEIRE - the Conference on European Industrial Requirements Engineering. Why wasn't it called RE-Day 2? Well, because it is ***two*** days long this time (19-20th October), with even more workshops, tutorials, industrial presentations, a tools exhibition and many other goodies to choose from! We're delighted to have Michael Jackson as Keynote Speaker, and he will be joined by a distinguished line-up of internationally-renowned experts who will give presentations on different requirements engineering topics. You really MUST be there!!

You should all by now have received your two year membership renewal packs from Membership Secretary Sara Jones. Please, process these and return them to her with your payment as soon as possible - to be sure you do not to miss the next issue of this newsletter (and the numerous other benefits of membership)!

Finally, I would like to welcome Dr Laurence Brooks onto the RESG committee as WWW Officer. Laurence is a Lecturer at the University of York, and brings with him expertise in both requirements engineering and human-computer interaction. So, make sure you bookmark the RESG web page (http://www.cs.york.ac.uk/bcs/resg/), and refer back to it often - changes are ahead!

Bashar Nuseibeh,
Imperial College, London

RE-Treats

Forthcoming events organised by the group.

Conference on European Industrial Requirements Engineering (CEIRE'98)

Date : 19th - 20th October 1998
Location: Novotel, Hammersmith, London.
Format: Two-day industrial conference, organised by the BCS RESG in association with RENOIR.

CEIRE'98 is the second (following RE Day, held in London on the 30th September 1997) in a series of European industry-oriented requirements engineering symposia organised by the BCS RESG in association with Renoir.

CEIRE'98 will provide a 2-day forum for the dissemination of experience and practice in RE between European software and systems engineering organisations and researchers. The programme will include tutorials and workshops on issues relevant to RE. It will also include a special track of papers giving the industrial perspective on RE. Papers are invited from industry or business organisations which focus on the practice of RE. Papers are particularly encouraged on, though not restricted to, the following:

* Examples of pressing RE problems faced by industry.

* Case studies of the application of particular RE techniques.

* Reports of good RE practice.

* Reports of the integration of RE into quality management and software process improvement programmes.

* Reports of issues concerning RE in specialist domains - e.g. dependable systems.

* Reports of how changes to the business or competitive environment impacts on RE.

Submission procedure:

Papers should be in English and of no more than 3000 words in length. They should be submitted by the 29th June. Authors will be notified of whether their paper has been accepted by the 10th August. Accepted papers will be published in the workshop proceedings.

Address for submissions:

Pete Sawyer
CEIRE'98 Industrial Programme Chair
Computing Department
Lancaster University
Lancaster
U.K. LA1 4YR
tel: 44 1524 593780
e-mail: Sawyer@comp.lancs.ac.uk

Programme committee:

Ian Alexander
Wolfgang Emmerich (coordinator)
Galal Galal (workshop organisation)
Orlena Gotel (exhibition organiser)
Sara Jones (tutorial organisation)
Gerald Kotonya
Neil Maiden (treasurer & local organisation)
Shailey Minocha
Cornelius Ncube
Bashar Nuseibeh (chair)
Pete Sawyer (industrial programme chair)
George Spanoudakis

Special industrial advisory committee:

Pere Botella
Sjaak Brinkkemper
Silvana Castano
Eric Dubois
George Grosz
Andreas Opdahl

Summary of important dates:

Conference dates: 19th - 20th October 1998
Submission deadline: 29th June 1998
Notification of acceptance: 10th August 1998

Distributed Requirements Engineering

November 98
Details to be announced

RE-News

Jackson Wins Lovelace Medal

We are delighted to announce that Mr. Michael Jackson will be the first recipient of the BCS Lovelace Medal for outstanding contribution to infomation systems development. Jackson was nominated by the BCS Requirements Engineering Speciaist Group, and his nomination was supported by BCS specialist groups on Software Reuse, Information Systems Methodologies (ISM), Object-Oriented Programming and Systems (OOPS), Formal Aspects of Computing (FACS), and Computer-Aided Software Engineering (CASE).

The award will be presented by the President of the BCS at the Society's Annual Dinner, to be held at the Brewery in London on 25 November 1998.

RESG Web address

Please note that the web address of the RESG is now: http://www.cs.york.ac.uk/bcs/resg/

Don't forget to update the bookmarks in your browsers!

Architectural Approaches To Software Engineering: Frameworks, Patterns, Components and Architectures

A Two-Day Seminar, 8th & 9th December 1998 at The Open University, Milton Keynes, UK Speakers will include

Alan Cameron Wills, Trireme International Ltd.
Bruce Anderson, European Object Technology Practice, IBM UK Ltd.
Hans-Albrecht Schmid, FH Konstanz University of Applied Science
Jan Bosch, University of Karlskrona
Oscar Nierstrasz, University of Berne
Steve Cook, European Object Technology Practice, IBM UK Ltd
Wolfgang Pree, Johannes Kepler University Linz

This seminar will gather renowned speakers from Europe with important research in the area of Architectural Approaches to Software Engineering. It will be covering topics such as: Frameworks, Patterns, Architectures and Components.

Who should attend? Practitioners and academics with interest in the new developments within software engineering and object-oriented development. We will encourage a lively exchange of ideas in what we believe to be an extremely important area of research.

For details see:
http://mcs.open.ac.uk/SoftEngSem/SoftEngSem.html

£150 (if the payment is received by the 30th September 1998), otherwise £180. Registration includes a buffet lunch and refreshments on both days

Praxis offers a course in Reveal

On the 14th-16th October 1998, Praxis Critical Systems are running a 3-day public course in Requirements Engineering. The course teaches the method, called Reveal, used by Praxis Critical Systems in its own Requirements Engineering work and is based upon currently available good-practice.

The course is aimed at Project Managers, Project Engineers, Systems Procurement Managers and includes bound copies of course notes, the Reveal process model, exercises in practical examples, CPD Accreditation, Reveal Course Certificate, Lunch and Refreshments

The course uses real-life examples of Reveal techniques resolving requirements issues that traditional techniques would have readily missed, causing massive re-work.

The course costs £895 plus VAT per attendee for the 3-day course and is held at our training facilities in the beautiful city of Bath, England.

The course will be lead by Anthony Hall and Andrew Vickers. For further information about the training course, please contact: Karen Green or Amanda Kingscote, email: info@praxis-cs.co.uk, tel: +44 (0)1225 466991

Calendar

October 1998

The Third International Conference on Practical Software Quality Techniques (PSQT'98), St. Paul, Minneapolis October 5-7, 1998.
http://tcqaa.org/psqt/index.html

International Workshop On Current Trends In Applied Formal Methods, Boppard, Germany, 7-9 October, 1998.
http://www.dfki.de/vse/fm-trends/

13th IEEE International Conference on Automated Software Engineering (ASE'98), October 13-16, 1998, Honolulu, Hawaii, USA
http://www.ics.uci.edu/~ase98

Workshop on Industrial-strength Formal specification Techniques (WIFT'98), Boca Raton, Florida USA, October 21-24, 1998.
Info: chengb@cps.msu.edu

November 1998

Sixth International Symposium on the Foundations of Software Engineering (FSE-6), Orlando, Florida, USA, November 1-5, 1998
http://www.ics.uci.edu/fse6

3rd IEEE* High-Assurance Systems Engineering Symposium, Washington, DC, November 13-14, 1998
http://kel.eecs.uic.edu/hase98/

17th International Conference on Conceptual Modeling (ER'98), Singapore, Nov 16-19, 1998
http://www.iscs.nus.edu.sg/~er98

METRICS 98: Fifth International Symposium on Software Metrics, Bethesda, Maryland, November 19-21, 1998.
http://aaron.cs.umd.edu/metrics98

December 1998'

Asia Pacific Software Engineering Conference 1998, Taipei December 1-4
http://www.selab.org.tw/apsec98

Software Process Improvement 98 (SPI98), Monte Carlo, 1-4 December 1998.
http://www.unipaderborn.de/cs/SPI98

The 19th IEEE Real-Time Systems Symposium, Madrid, Spain, December 2-4, 1998
http://www.cs.umd.edu/~rich/rtss98/

Second IEEE International Conference on Formal Engineering Methods (ICFEM'98), Brisbane, Australia, December 9-11 1998.
http://svrc.it.uq.edu.au/icfem98/

5th Maghrebian Conference on Software Engineering and Artificial Intelligence, Tunis, Tunisia, December 8-10, 1998
http://www.irsit.rnrt/

January 1999

Data Semantics 8 (DS-8): Semantic Issues in Multimedia Systems, Rotorua, New Zealand, 5-8 January 1999.
http://zulu.cs.rmit.edu.au/~ds8

February 1999

Third IFIP International Conference on Formal Methods for Open Object-based Distributed Systems (FMOODS'99), Florence, Italy, February 15th - 18th, 1999.
http://www.dsi.unifi.it/fmoods/

The First Working IFIP Conference on Software Architecture (WICSA1), San Antonio, Texas, USA, February 22-24, 1999.
http://www.bell-labs.com/usr/dep/prof/wicsa1/

International Conference on Process Modelling, Cottbus, Germany, 22-24 Feb 1999.
http://www.processmodelling.tu-cottbus.de

September 1999

FM'99: Formal Methods 1999, The World Congress on Formal Methods in the Development of Computing Systems, Toulouse, France, Late September 1999.
http://www.it.dtu.dk/~db/fm99/

RE-Readings

Reviews of recent Requirements Engineering events.

How to Use Scenarios and Use Cases in the Systems Development Process

City University, 14th May 1998

Report by Ian Alexander

The eagerly-awaited one-day event, entitled '**How to Use Scenarios and Use Cases in the Systems Development Process**', was the RESG's first venture into scenarios. It was a sell-out success, and the treasurer could be overheard muttering about hiring the Albert Hall instead of City University's Senate Suite. The hall was packed and the lucky participants were able to enjoy a wide range of excellent talks from both business and academic speakers. **Neil Maiden** expertly organised and controlled the proceedings throughout the day.

Andrew Daw of GEC-Marconi spoke on 'Scenarios and their Application for Requirements Definition'. He began by saying that the only document owned by the customer (the MoD) was the requirements database; all the models belonged to the developers. The requirements were therefore crucial for controlling a project. Using the example of a naval scenario involving a ship under threat from attacking aircraft and a submarine, he showed how the scenario could be modelled (rather as on a radar-screen) to present a series of questions about the desired capabilities and effectiveness of the future system (such as a warship). From these functional requirements (e.g. 'detect airborne threats at x miles') came the question 'what resources and support do I want to achieve that?'. This led naturally to constraints such as resources, costs, and performance. The approach led to successive baselines, from the perceived shortfall using today's ships, to an ideal baseline, to a realistic and affordable one. Then attention could move on to structured design (he apologised), architectural modelling and simulation, and hence to MMI and Human Factors modelling. Scenarios were central to the whole of this development process. He agreed with Prof. Ken Jackson

(from the floor) that his Scenarios closely resembled the military 'Concept of Operations'.

Ian Graham of Chase Manhattan Bank spoke trenchantly about 'Validating Requirements - Beyond UML!'. He began by asking 'Guess what the U stands for, or pick your favourite!' He argued that there were serious structural and semantic weaknesses in UML, and that it was programmer- rather than business-oriented. The language used bi-directional associations and lacked proper rulesets; the sequence diagrams did not allow for concurrency (parallelism), and the sequence diagrams were introduced too early. The various types of diagrams were not integrated. Use Cases were also inadequate for business modelling, as they were not bona fide objects, they were too numerous, and they were not generic, atomic, or goal-oriented. He pointed to SOMA, BOM, and Syntropy as better approaches.

Nicola Millard of BT Laboratories spoke on 'Once upon a time … Using Scenario-Based Techniques Developed to Elicit Requirements from Children with Adults'. Starting with a photograph of a child -- herself, Millard explained the challenge from the Human Factors point of view, and, supported by a short video clip, demonstrated her own highly effective ways of getting useful technology suggestions from potential customers. The problem was that users' knowledge was tacit, their requirements were fuzzy, and the future was unknowable so there were in fact no real users. Environments were volatile, users did not know how systems might help, and above all they were not designers. With children, it was no good asking questions directly, while ethnographic observation, although useful, did not yield requirements. Mediated Interviewing, consisting of distracting the children with a task and then interviewing them while they did it (Harry McMahon, Ulster) was effective. Children were good at drawing, storytelling, and discussion. Surprisingly, the techniques carried over quite well to adults, though Millard had to start by breaking down their inhibitions. Brainstorming worked well with both age groups; storyboarding was interesting but no good for critical assessment, while modelling (with clay models, roleplay, and so on) was quite good all round. Storytelling (by one teenager) accompanied by roleplay (another teenager) generated heaps of requirements, all in the context of a believable future scenario.

After a refreshing cup of coffee accompanied by software demonstrations, **Ian Alexander**, an independent consultant, spoke under the title 'Task Models Elicit Requirements and Generate Scenarios'. The approach is to describe a business process as a typed hierarchy of tasks. A task can consist of a simple sequence of subtasks, or of subtasks which can run in parallel with each other, or of a set of alternatives. A model structured in this way can generate scenarios by tracing (animating) paths through the hierarchy, and can be used to organise requirements and generate test scripts. Tool support for the approach was illustrated with examples (details can be found on http://www.scenarioplus.com). From the floor, Ian Graham pointed out that one could generate use cases and UML diagrams from such a task model.

Ian Spence of Rational Software Corp., spoke on 'Use Cases: a Reusable Way to Represent Requirements'. He had a difficult path to pursue after the various assaults on UML, beginning 'I'm actually going to talk about Use Cases'. He argued that one needed to start by defining a system boundary (rather than by elucidating this while defining the requirements), and that system/software requirements did not adequately describe functionality or the needs of documentation. Use cases implicitly traced back to requirements, but had to be accompanied by a glossary which 'acts as a model of the business entities not captured by the use case model'. He granted that there was an infinite number of scenarios, but stated that there was in contrast a finite number of flows of events within the context of a use case. Use cases were stated to be 'performed by a set of interacting classes' -- though this presumably only applies when the case results in software. Tests however had to be based on the scenarios which came from the use cases. Spence was closely questioned from the floor: one user said: 'I'm still confused about how to use scenarios'; another, asking what to do about his scenario explosion, was told 'That is by using flows not scenarios'. It seemed that speaker and audience had different languages.

Lunch offered an opportunity for the participants to question the speakers closely over some excellent food, and for the various software tools and prototypes to be shown off. The afternoon was devoted to the academic research now under way on Scenarios in the impressive **CREWS** [Co-operative Requirements Engineering With Scenarios] project (further details at http://sunsite.informatik.rwth-aachen.de/CREWS/index.htm).

Matthias Jarke of RWTH Aachen, Germany, spoke on 'CREWS, A European Project on Scenarios'. CREWS had classified scenarios into an effective framework. 15 real projects had been selected, of widely differing types including air traffic control, process engineering, banking and medical information. The framework allowed for system, usage, subject domain, and development process 'worlds', as well as for social, technical, and cognitive 'dimensions'. The focus was very much on concrete reality rather than the academic and abstract methods and tools debate. Each scenario could be classified by asking about its contents (knowledge), purpose, form (in which it was expressed), and lifecycle (how it was manipulated). The findings from industry were surprising: structured text was heavily used, as were diagrams: natural language text and images less so, while none of the 15 projects was using animation or simulation. Three main types of context were used: scenarios purely internal to a system, with no context at all; interaction scenarios, looking at direct interactions between actors and systems; and environmental scenarios, looking at all of the above, together with the system environment. The main use of scenarios (13 / 15 projects) was to refine abstract conceptual models, to reduce complexity (all of them), and to reach towards agreement and consistency (all). Scenarios complemented prototypes

and helped to define static (object) models. In other words, they elicited functional requirements.

Colette Rolland of the Sorbonne, France, spoke on 'Guiding Goal Modelling Using Scenarios'. The approach was to guide and evaluate the process of writing scenarios. The Ecritoire tool contained rules for writing clear, complete requirements and helped authors towards expressing goals accurately. Users write scenarios (in English) following style and content guidelines. Rules in the tool parse the text, and help to extract a complete and unambiguous scenario, which the tool puts into structured text such as 'If the card is valid, then a prompt for code is displayed by the ATM to the user'. The author interacts with the tool to fill in gaps in the requirements.

Klaus Pohl of RWTH Aachen, Germany, spoke on 'Requirements Elicitation with Real World Scenes'. He began by saying that 'in industry nobody knows what their current system is doing', so it was useful to analyse existing systems and find out which requirements they actually satisfied. There was little point in making sophisticated models if the users would not understand these, so the project was aiming to describe scenes using rich media including video, images, and speech. A scenario could then be played out by stepping through these. A tool had been made to mark up extracts of videos and describe them: did they show the goal being reached or not, were they for reference. A system of cross-links enabled users to look for relevant examples and for those that indicated successful execution.

Alistair Sutcliffe of City's own Centre for HCI Design spoke on 'Scenario-Based Requirements Validation'. CREWS' SAVRE tool deals with the problem of exceptions (such as human mistakes caused by systems), looking at human factors within models of the enterprise through a scenario-based approach. Scenarios could be seen as stories, captured from human experience; as a sequence of events, created by tracing through a use case; or an actual example of use of a system. By generating possible scenarios to explore the implications of a use case, one can discover where, when, and why things go wrong, and perhaps suggest possible fixes (in the shape of new requirements). Attention is thus focussed on the exceptions. On the way into a system, users may fail to supply needed input; on the way out, the system may fail to supply output (at the right time, to the right place). Thus the exceptions point to both usability and system requirement problems. SAVRE includes guidelines, helps with capturing use cases (the Sorbonne work), generates scenarios semi-automatically, and prompts for exception cases. There is also an interesting attempt to reuse 'generic' requirements, offering advice in a timely way. CREWS thus attacks the central problems of requirements engineering on several fronts.

Lastly, **Patrick Heymans** of the University of Namur spoke on 'Animation of Scenarios for Requirements Validation'. The approach uses an interactive tool to animate low-level industrial process scenarios with tight time constraints (seconds). Users are helped to discover errors in specifications, and receive feedback in the vocabulary used in the specification. Animation is distributed, so that several stakeholders in the specification collaborate to step through a scenario. Interaction allows non-determinism to be handled (as with Ian Alexander's approach) by allowing users to make choices; the effect is a game-like simulation of the future system.

The day closed with a panel discussion, at which there was quite a lively debate with questions from the floor about what exactly a scenario and a use case might be. Ian Graham said there were as many definitions as speakers at a previous conference he had attended, and that even in Jacobson's book there were two conflicting definitions of use case! But he boldly ventured a definition.

Those members - and non-members - who missed Scenarios Day can console themselves with the forthcoming RESG conference, CEIRE, which amongst many other good things offers a strong line in Use Cases and Scenarios. CEIRE promises to be our best event ever. Book early to avoid disappointment!

Managing Requirements Change: A Business Process Re-Engineering Perspective

Joint IEE / RESG Colloquium, Savoy Place, 11th June 1998.

Report by Ian Alexander

On Thursday the 11th of June, the IEE professional group A1 (Software Engineering) held a joint event with the RESG at the IEE, Savoy Place, London. The event was sponsored by the 'Systems Engineering for Business Process Change' (SEBPC) project, a major research programme on this topic. The meeting was chaired by Dr. Bashar Nuseibeh (RESG) and Prof. Peter Henderson (Southampton, for the IEE).

Richard Veryard, an independent consultant, spoke first, on the provocative subject *'Demanding Change: how to remain in business despite IT'*. He began with a metaphor for different paces of change: older children in a playground want to whiz round, whereas toddlers and their parents do not. In business, some entities are more able to stay on the roundabout than others. Humans are currently rushing along, while other species are being flung off into extinction. The carmaker Toyota is setting a fast pace for its rivals. We value speed of change and flexibility in business: but IT's speed does not equate to business flexibility.

There is unhappily a poor correlation between IT spending and business profitability. 'It may be simply because profitable companies have more money to waste on IT.' Veryard went on 'but let us suppose that all this wastage can be attributed to poor software practices. Successful IT projects perfectly meet their requirements' (laughter): they adapt the business ideally to the situation now, in a formalized and highly specialized way. The effect is stiffness in adapting to change: 'adaptation decreases adaptability'.

Business change their names (ICI … Zeneca), their specialisations (Xerox not primarily about photocopying), their size and position (IBM), their nationality (ICL, Rolls-Royce), their staff... What survives? Something has, but it certainly isn't easy to identify.

Businesses have many competing descriptions. The official, formal one of the organigrams may be a far cry from what really happens. 'Informally and covertly, the effective organization may nevertheless change. Formal reporting lines are bypassed, managers take more initiative than they are officially permitted, people find ways of evading the more bureaucratic controls. The level of awareness of these informal changes may be very variable. Politically naïve participants may only gradually become aware of discrepancies in the formal account. Politically astute managers may .. plan consciously for their own benefit .. and may attempt to deliberately distort the way the organizational structure is perceived by others.'

IT flexibility needs to be aligned to (real) business flexibility: not vaguely, but firmly committed to 'generic systems and components which will satisfy a wide range of descriptions'. Requirements must change within this framework.

Bob Snowdon, of Teamware (formerly part of ICL), spoke on 'the Role of process based IT in supporting business change'. Businesses used to be considered as a series of separate functions (machining, polishing, etc), each with its own IT support which tried to optimise the function in isolation. The more modern outlook is that processes exist (horizontally) across the functions, so process-based IT needs to support the whole set in a unified way.

A real danger is that once a process has been identified and supported, it is effectively frozen into place: 'the use of such IT can ossify the work practices they support and make it hard to change a business'. In other words, optimising a business for its current processes may prevent it from adapting itself afterwards. Wealthy firms which spent heavily on IT are now 'lost in electronic concrete'. The danger may be reduced by trying to support all parts of a business with a common underlying system (allowing processes to be rearranged without complete re-engineering, next time round). This would mean supporting actual operations (machining, etc) with the same system as operations management (which monitors and controls operations). In other words, a business process and the process that controls it would be integrated.

Snowdon concludes that 'finally there is the problem of determining how the proposed change may be brought about. This is, of course, a process again…' The satisfying but sobering conclusion is that the BPR consultant and requirements engineer are part of the process of change, so we cannot sit back and study the process as an isolated abstraction. BPR and RE are soft, and complex, systems. Perhaps they can never be fully automated.

Malcolm Munro of Durham spoke on '*Software Evolution Research at the Centre for Software Maintenance*'. The CSM (http://www.dur.ac.uk/CSM) was established at the University of Durham in 1987, the first in the world to focus on maintenance. The root of the problem is expressed in the laws named for Manny Lehman, who stated that

1. Software must continually evolve, or grow useless, and
2. The structure of evolving software tends to degrade.

Much old 'legacy' code in (for instance) COBOL is both weakly-structured and vital to business. The CSM has grappled with this stuff by making static analysis tools which draw graphs and trees to represent program structures visually, and by developing a method in the IDENT project to reverse engineer legacy systems.

The RELEASE project is trying to help the migration to 'more appropriate software structure' – whatever that might be (easier said than done). RELEASE builds on the IDENT work by adding modern techniques including drawing dominance trees (by discovering dominant nodes), data slicing and data clustering.

Finally, the new FLEXX project aims to define what highly flexible software (not brittle in the Lehman sense) might look like.

Richard Veryard observed that it would be easiest to discover the causes of inflexibilities: being positive (to shape better systems) would be harder.

Rob Pooley and **Perdita Stevens** of Edinburgh were unfortunately unable to attend. Their paper discusses the use of patterns in re-engineering: such patterns differ from purely design patterns in addressing the process, not only the end-point. Happily there is a website, http://www.dcs.ed.ac.uk/home/pxs/reengineering-patterns.html, which presents their candidate patterns. The authors welcome comments and proposals on the subject.

John Edwards of Loughborough spoke on '*Agile System Design and Build*'. It is a truism that requirements change rapidly. The BPR of the 1990s has been less successful than was trumpeted, mainly because IT systems are hard to change. He described 2 jobs, one being tested for a City bank and one just completed on a practical method in which system designs are translated 'in one step' to executable code – in effect the designs are executed by a Petri Net modelling engine.

The bank application is to support discretionary management of clients' funds (by buying and selling bonds, etc). The project analysed the requirements as usual, but found that 'the analysis looked only at the process details at a single moment', whereas in fact the application 'has a significant dynamic component', and 'what is required is greater IT agility which could empower the dealer to modify his application to suit his [current] business process'.

This is a startling finding. Apparently the dealers are typically physicists and mathematicians expert in calculus, and they carry with them subtle spreadsheet models which

the banks would much like to control. The aim of designing an 'agile' system to manage huge amounts of money, safely, in a risky environment, with staff who will leave if pressured, and code which can be modified dynamically, is certainly ambitious. Clearly the users must be able to specify the system at a high level. Edwards identifies 5 'means' required for such an agile system:

1. Means to create design models
2. Means to execute (conventional) component functionality
3. Means to execute the behavioural model
4. Means to map models into an integration mechanism
5. Means to generate and update executable models.

Edwards claims that the 'separation of function, behaviour, and interaction is established during design and maintained throughout system implementation and operation'. Whether this will enable users to create their own systems, time will tell.

D.W.Bustard of Ulster spoke on '*BPR through SSM: An Incremental Approach*'. Soft systems methodology (Checkland, *SSM in Action*, Wiley, 1990) offers a way of developing a vision that can guide a business through the risky process of re-engineering. He described his approach which uses an 'evolutionary system plan' involving a series of steps, determined by cost/benefit and risk analysis, and documented in a set of soft system models. His RIPPLE project (http://www.infc.ulst.ac.uk/project/150) is part of the SEBPC programme.

There is a danger inherent in a technological worldview, which is that novelty and 'cleverness' are encouraged, at the expense of practical, goal-driven change. Early BPR (e.g. Michael Hammer, 1990) argued for sudden revolution, 'but the associated cost, disruption and risk may be unacceptable'. More modest phased approaches may do better. Bustard presented a cycle consisting of assessing risks in the initial system, assessing risks in the target system, defining the changes (by subtraction between the two models), analysing costs and benefits, and (re-)planning accordingly. The plan contains prioritised changes with means of controlling the risks. The cycle can be repeated several times. A key concept is that in each cycle, the business and its computing support evolve together (in small steps) so that they remain aligned with each other. Co-evolution is a Darwinian concept with wide application.

Eve Mitleton-Kelly of the London School of Economics (http://www.lse.ac.uk/lse/complex) spoke controversially on '*Complexity: Partial Support for BPR?*'. She is working with concepts of adaptation, co-evolution and complexity, derived from evolutionary biology (which itself makes use of non-linear dynamics or Chaos theory). Due to an accident with the photocopying, her paper itself appeared intermingled with her slides in a highly stimulating but non-linear form!

There is no single theory of complexity: it draws on physics, chemistry, biology, mathematics and even economics. It is strongly related to BPR, but so far there has been very little work with a social systems context. The idea of adaptation to an environment is persuasive, but is too limited: for the environment (other species, competing firms) adapts to your adaptation, so what is really happening is a co-evolution of each system *with* its environment, its ecosystem. In other words, the idea that you can, once, perform a BPR operation is inherently misguided. The BPR is itself a stage in the process with influences on any number of other systems. A strong conclusion is that complex systems behave in ways that do not have determined outcomes.

A business can be seen as a 'dissipative structure' (like an organism) which is an open system exchanging energy, matter, and information with its environment. Change is inherently unpredictable. Attempts to change such a structure, such as BPR, are 'disturbances which push it into a higher-energy (away from equilibrium) state, which is harder to sustain'. Successful BPR may cause a disintegration into instability, or a leap to a new level of organisation. Innovation, in other words, is a risky but possibly stimulating move away from an old state of balance. Far-from-equilibrium systems 'are forced to experiment and explore their space of possibilities, and this exploration helps them discover and create new patterns of relationships and ... structures'. BPR thus operates on the 'edge of chaos'.

A related viewpoint (Gregoire Nicolis) sees such creativity as the spontaneous emergence of new behaviours, structures, and patterns. Systems are self-organising, not designed – provided change is not blocked by interventions such as BPR. Over-zealous restructuring creates possibly-helpful turbulence, but fails to allow new connections to form (like moving plants around the garden before they can take root?). Change cannot be strictly channelled, but must find its own ways. After restructuring, a business needs time to establish new ways of working.

Complexity does not provide a complete and coherent methodology, but it does offer a way of thinking about the world of business, a new way of seeing. Some possible approaches are to encourage businesses to learn how to recognise new emergent patterns, and to co-evolve with other organisations.

It seemed to me that there was a striking unity of theme in these talks. If the parody of a colloquium is one speaker talking on neutrinos, one on cytoplasm and one on how to improve the grinding of valves, this was exactly the opposite. The key point seems to be that system change always risks creating brittleness, an inability to adapt to later changes. In Darwinian terms, many adaptations reduce adaptability: for instance, Bee Orchids have a flower whose lower lip closely and beautifully resembles the abdomen of a small bee: pollination is effected when the bee attempts to mate with the bee-image. The flower has specialised to the point where it depends absolutely on the bee for its survival, which is now threatened.

On the other hand, some adaptations plainly favour adaptability: the human hand, the vertebrate eye, and (par excellence) the human brain are structures which are effectively 'designed' to operate in an unpredictable environment for a myriad of purposes.

Isn't the conclusion, then, the subversive agenda that less IT is often better?

- IT can and often does cause deskilling;
- IT dangerously reduces co-operative and self-organising work
- IT reduces the ability of people to react to change.

Change should not come from outside a business, but from within. The role of the consultant or requirements engineer should surely be to co-operate with their clients, teaching them how to adapt, with the help where necessary of new technologies. If, as Bertrand Meyer said, 'the role of a trainer or consultant is to empower the customer', then our goal must be to help them cope with change, and at all costs to avoid locking them into legacy systems. The role of BPR is to enhance adaptability.

RE-Papers

Why You Need Requirements Engineering

by Ian F. Alexander

Introduction: Why Projects Fail So Often

The reason why requirements engineering has come into being as a discipline is simple: projects fail, horribly often. In a famous report (Standish Group, 1995; see also Scientific American, September 1994), the stark results in big business were bleakly presented. 53% of projects investigated had failed; a further 31% were partially successful, requiring some rework or extension of time and budget for completion. In fact, on those two crucial measures, project planning had failed also: costs overran by 89% on average (some projects were much worse) and planned time was overrun by 122%.

It is unfortunately all too easy to give examples of large and famous projects that were abandoned, were delivered late, or had greatly reduced functionality, resulting in large costs or even loss of human life. Aerospace projects are perhaps especially conspicuous: the first Ariane V launcher, a broad, stubby vehicle, was lost because a software module designed for the tall, slender Ariane IV was included without re-testing "because it had been tested already". The module controlled the pointing of the launcher, and had been necessary in the earlier design because of its sensitivity to small deviations from its intended course. In fact, Ariane V did not require such a controller at all. Worse, the control laws embodied in the software matched the flight dynamics of Ariane IV, and were entirely wrong for the new design. The launcher veered off course soon after lift-off, and exploded (destroying the entire set of CLUSTER solar exploration mission satellites). The spokesman explained on the television news that "it was only a software issue" and therefore nothing serious.

Why do corporate planners and managers do so badly? The unequivocal answer from Standish was that people had not specified what they wanted properly. In fact, 13.1% of the projects failed because of incomplete requirements; 12.4% failed through lack of user involvement (a key element of requirements engineering); 10.6% through inadequate resourcing; 9.9% from unrealistic (customer) expectations; 9.3% from lack of management support; 8.7% because the requirements kept on changing; 8.1% through inadequate planning; and 7.5% because the product, when finally delivered, was no longer needed.

It is striking that among this depressing catalogue of disasters, there is no mention of all those teenage-programmer anxieties (such as: will I be able to handle the Java class library?). Technical skill was simply not the issue: the problems of real projects lay with requirements, management, and boardroom politics. Everyone can see that if the reason why your organisation is slow to respond to customer orders is that Sales take a week to sign each document, then installing a faster network will make no difference. It is curious, then, that in the face of clear evidence that the deepest problem for large systems is to improve requirements management, the solution still proffered by governments is to try to train more people in technical skills.

What needs to be done

The diagnosis, then, is mismanagement, especially of requirements. Whenever project decisions are taken other than on the basis of a clear assessment of need, mistakes will happen.

The first step is to find out what results the users of a system want. There are plenty of opportunities for system and software analysts, designers, and programmers to forget this during a project. Most of them are unlikely to meet real users, so the only chance of getting the message across is to write down the user requirements, and to make sure that the users know that they want those requirements to be implemented. Then, when the developers start to plead time, or complexity, or budget, the users' representatives can check the specification and insist on getting what they wanted.

After that, the sole aim of requirements engineering is to keep to the specification through thick or thin. The basic conceptual tool for this purpose has the ungainly title of Traceability. This means nothing more than making sure that each requirement is satisfied by the design, and (in the other

direction), that each element of the design is called for by the requirements.

Both of these conditions are potentially one-to-many relationships, so the general case for traceability is a many-to-many problem. It is easy enough for a tool to show that, for instance, the requirement that the driver of a car be warned of any dangerous state in the engine has called out the design of the oil pressure warning light; but that design element in no way satisfies the whole requirement: several other traces must be added, probably by hand. Traceability tools, then, are unlikely to put many designers out of business.

How to Engineer your Requirements Better

Navigation is said to be the art of proceeding, via a known path, from a known position to another known position. System engineering is no different: everything depends on process.

The essential insight of requirements engineering is that success depends on looking back (to the project's goals and requirements) as well as forward (to the next deadline).

The basic process is to document first the problem that is to be solved, and to make sure that the users 'own' those requirements.

Then, you trace those to system requirements (specifying the solution).

Finally, the traces lead to a system architecture, which can be shown to satisfy all of the requirements on it. Verification mirrors the whole process.

That architecture can in turn be broken down into subsystems, and contracts (containing subsystem requirements) can be placed for those to be procured or made. Incidentally, the subsystem requirements have to trace to the architecture which created those subsystems; attempts to trace directly back to the system requirements invariably mix up design with specification, with predictable results.

Process and System Modelling

All of this comes down to modelling processes and systems in appropriate ways. For example, a system could be an aircraft, and the many processes involving it might include its development, qualification, commercial use, and modification. Requirements engineering can begin by modelling the required tasks that a system will have to perform, and then generating some key scenarios to show how the system will be used. These can cover contingencies as well as normal operations. But different practitioners use many different approaches.

For your project, you need to think through an information model that will suit it, and then implement that with a process that will make the project's information work for you -- and for your clients.

Learning how to do this takes time: contact with other practitioners and users, such as through the BCS Requirements Engineering Specialist Group (RESG), is a good way to start.

Requirements Tools

A tool is virtually essential to support a disciplined system engineering approach. The tool must represent the structure of specifications, with a hierarchy to organise requirements into understandable documents. Each requirement must be a record in its own right, with any number of attributes such as risk, cost, priority, and review status, according to your project standards. And crucially, each requirement must be linked securely to the objects that trace to or from it. Specialised processes, such as requirements acquisition, task modelling, scenario generation, model validation, or rapid applications development can also be assisted greatly by the right choice of tools. Researchers and practitioners regularly present their latest thinking at RESG seminars, tutorials, and workshops: see below for details.

Numerous requirements tools are now available, but few of them actually meet these requirements-on-requirements. It is not really satisfactory to use (or create) a tool based on a simple relational database: while such a tool can manage the idea of attributes well, and can handle traces using a table of relations, it is not well suited to representing a specification as a document, or to diagramming system models or document structures. The best commercial tools use a database which is both inherently hierarchical and provides links. You may also want specific interfaces to CASE or other tools, or to be able to build in your own extensions by programming.

The exhibition presented with the CEIRE conference gives you the opportunity to meet all the major vendors and to try out their products for yourself. Details of CEIRE are given below.

Within the BCS: The Requirements Engineering Specialist Group (RESG)

The BCS has been active for several years now in encouraging academics, industrialists and practitioners to get together to address some of the issues discussed here. The Requirements Engineering Specialist Group (RESG) actively explores a wide range of aspects, from requirements elicitation and modelling, use cases and scenarios, through analysis and specification writing to special aspects such as identifying and reusing patterns in requirements. The group is based in London, but has members all over the UK and seeks to hold meetings wherever there is sufficient interest.

The Conference On European Industrial Requirements Engineering (CEIRE)

The next Conference On European Industrial Requirements Engineering (CEIRE), jointly sponsored by the RESG and the European Union's RENOIR project, will be held on the 19th - 20th October 1998. The conference will focus on actual practice and experience in the field. Hope to see you there.

REFSQ'98 Workshop Summary

by Andreas L. Opdahl & Klaus Pohl

The workshop proceedings can be ordered by email from pohl@informatik.rwth-aachen.de (cost: £20)

The Fourth International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'98) was held in conjunction with CAiSE*98 in Pisa, Italy on June 8th and 9th 1998. The workshop was organised by Eric Dubois, Andreas L. Opdahl and Klaus Pohl.

In the following we give an overview of the presentations and of the numerous fruitful discussions that took place during the workshop.

Introduction and Workshop Structure

Achieving high quality means to realise the needs of users and customers. These needs are elicited, negotiated, documented and validated during requirements engineering (RE). RE is therefore the most crucial phase of software development. The goal of RE is not just to produce a specification that is as consistent and complete as possible: RE is also a cooperative and iterative learning process. According to Pohl [2] the goal of RE is to produce a traceable requirements specification that is as complete as possible and that is documented using various appropriate representation formats, and to establish a sufficient common understanding among the stakeholders involved. Conventional RE methods normally support only parts of this process or help stating only specific kinds of requirements.

The goal of the REFSQ workshop series, which has been held in conjunction with the Conference on Advanced information Systems Engineering (CAiSE) since 1994, is to improve the understanding of the relations between requirements engineering and software quality. More precisely, REFSQ aims at having intensive discussion provoked by brief presentations about:

- Solutions to known RE problems and shortcomings;
- Innovative research ideas possibly initiating new research directions;
- Industrial problem statements ;
- Generalisations from individual industrial experiences.

In response to the Call for Papers for REFSQ '98 which addressed the above problems we received 31 submissions. After the reviewing process we accepted 12 full papers of high-quality, five position/preliminary result papers, and four industrial problem statements. Consequently, the workshop comprised 21 presentations with around two-thirds of the workshop time set aside for discussion of individual papers as well as general issues.

With respect to the described research problems and the contents of the accepted papers, we divided the workshop into four full-paper sessions: Goals in RE (chair: Andreas Opdahl); the RE process (chair: Klaus Pohl); specification and validation (chair: Philip Morris); scenarios in RE (chair: Eric Yu). In addition, there were two sessions of position statements and preliminary results chaired by Neil Maiden, as well as a session of industrial problem statements chaired by Klaus Pohl.

To ensure the effectiveness of the workshop a full paper presentation was restricted to 15 minutes and followed by a 20 minutes presentation-related discussion. Each discussion was initiated by the two other speakers of the session acting as discussants. At the end of each session, the major topics raised by the talks or the talk related discussions were elaborated. The results of the discussions are summarised in the session summaries (see below). Presentations of position papers and preliminary results were restricted to 10 minutes with 10 minutes set aside for discussion.

The presenters had to make the context of each talk explicit by drawing one or more labelled arrows between a given set of concepts. This set of concepts comprised people (or stakeholders) which participate in the software development process and products (or documents) established by the process. The arrows indicated the relations between the concepts addressed by the paper. While arrows between people indicate some kind of social interaction, an arrow between documents describes technical and logical relationships. Arrows between people and documents usually indicate production/usage relationships.

Each talk was summarised by the presenter and the discussants by answering the following questions:

- Which quality features are addressed by the paper?
- What is the main novelty or contribution of the paper?
- How will this novelty or contribution improve RE practice and/or RE research?
- What are the main problems with the contribution and/or with the paper?
- Can the proposed approach be expected to scale to real-life problems?

The workshop was closed by a general discussion which aimed at integrating the session summaries and identifying important research topics. This led to a final workshop summary and a research agenda of topics worth elaborating further to make RE processes more efficient and effective, and thereby improve software quality in the future.

Session Summary: Goals in RE

Overview of Talks

This session continued along the lines of a similar session on Goals and Agents from REFSQ'97 [1]. The presentations in this session proposed to use goals as a high-level abstraction medium for structuring and abstracting the content of scenarios and requirements.

In the first presentation, Eric Yu (co-author: J. Mylopoulos) emphasised the increasing use of goals in RE. Eric characterised existing goal-based RE approaches. He argued that the increasing consideration of goals during RE requires a better integration of goals with the various others products

produced, e.g., specifications, viewpoints, argumentation, rationales, as well as with the RE methods and processes applied (see discussions for details). Eric suggested three major potential integration paths, namely ontological analysis, synthesis and/or the integration with existing RE frameworks. The problem with the proposal was the missing clear differentiation between goals and requirements. The question if industry should apply goal modelling techniques in addition to requirements modelling was controversially discussed.

The presentation by Peter Haumer (co-authors: K. Pohl and K. Weidenhaupt) focused on the improvement of the quality of current state (goal) models. Peter argued that important facets of the existing system should be captured using rich media, e.g. video. In addition, parts of the captured observations should be related to the goals of current state model which have been defined/modified based on the captured observation. Thereby, so called extended traceability is established. Among others, extended traceability improves the comparison of different observations, reviewing of conceptual models, explanation of the models to, e.g., new project members, and the discussion of the models with stakeholders not familiar with the modelling technique. In the paper related discussion the validation of the approach in real industrial projects was suggested. The integration of patterns, e.g. goal patterns, could support the elicitation of goals from the captured observation as well as the validation of goals against the observations.

The last presentation in this session by Camille Ben Achour (co-authors: C. Roland and C. Souveyet) proposed the organisation of large sets of (textual) scenarios using goal hierarchies. By assigning goals to scenarios and organising the goals using three types of relations (refine, and, or) a structure for managing scenarios is established. The refinements levels are thereby restricted to three possible layers which define system internal scenarios, system interaction scenarios, and contextual scenarios (from specific to more generic). The management and evolution of the resulting goal structure is supported by heuristics which, e.g., support the change or the introduction of a new goal in the hierarchy. In the discussion the comparison of the proposed goal structure to and-or refinement trees and other goal tree proposal was requested. As a major criticism, the organisation of goals in tree-structure was questioned. Experience seems to indicated that the relation between goals span more a net than a hierarchical (and-or) tree.

Discussion Summary

There was a general agreement that goals should be considered during RE. Among others, they could and should serve as basis for defining more detailed requirements, i.e. a goal should provide the rationale for introducing a (set of) requirements. As requirements, goals only exist in a given context. The introduction of goals in the RE process is especially helpful if the project at hand is somewhat fussy, the stakeholders involved are diverse, and/or if a complex interacting system is being developed. To empower the definition and use of goal models in RE, a much better integration of goals with existing RE models and RE modelling techniques/methods is required.

Existing goal modelling approaches are quite diverse. For example, the term goal is used to refer to different things, e.g., high-level business goal, performance goals, stakeholder goals, individual goals. Towards a common understanding a framework is needed which proposes a classification of the various kinds of goals. On the other hand, such a framework should define the relations between the various goal types and other RE artefacts, e.g. requirements and scenarios.

There was thus a general agreement that a reconciliation of existing goal oriented RE approaches is required. Among the participants there was no preference if such a reconciliation should be driven by the integration of goals with existing RE frameworks, by ontological analysis, by synthesis or by a combination of the three possibilities.

Session Summary: Position Papers and Preliminary Results 1

Overview of Talks

The presentation by Àngels Hernández (co-author: N. Castell) dealt with the derivation of requirements specifications from natural-language texts as realised in the SAREL system and with using SAREL to compare specifications. The discussion pointed to the problem of explicitly defending the cost of doing natural language analyses and to the need for explicit semantic and/or conceptual models underlying such an approach.

In his talk, Vincenzo Gervasi (co-authors: F. Fabbrini, M. Fusani, V. Gervasi, S. Gnesi and S. Ruggieri) presented a linguistic quality framework for natural-language requirements based on the framework of Krogstie, Lindland, Sindre and Sølvberg. Their framework comprised four quality types and 7 quality factors with associated quality criteria and supporting activities. The discussion raised the question of how much the application of such a framework would depend on domain knowledge and whether the framework would be useful for specifications that are not in natural language, but for instance in the form of goals or scenarios.

Bernd Deifel then discussed RE issues in the development of complex COTS using Pohl's classification of four main RE activities: elicitation, negotiation, specification and validation/verification. The main discussion topic was the extent to which - and in what ways - RE for COTS differed from RE for a single customer. Bernd also pointed out that a COTS product is not only a piece of software, but that it also has organisational implications in that it needs to be tailored to and implemented in the user organisation and that this should have consequences for how their requirements are established.

Session Summary: The RE Process

Overview of Talks

The session comprised three presentations which all reflected the increasing concern for process in addition to product issues within both RE and the broader systems engineering and software development communities. In the first talk, Peter Sawyer (co-authors: I. Sommerville and S. Viller) presented a "good practice guide" for RE developed in close co-operation with industry. A main driving force behind their work was the problem of disseminating RE research results to industry. In response, they have devised and described a set of 66 good practices and organised them into a 3-level maturity model for RE inspired by the capability maturity model for managing software development processes. Another discussion topic was whether research results can be disseminated in general form or if they must first be adapted to each specific application domain. The participants also questioned to what extent "good practices" can be identified and described across application domains and even between individual organisations.

In his talk, Wing Lam (co-authors: V. Shankararaman and S. Jones) also focused on "good practices", but with emphasis on management of changing requirements based on work within the DESIRE project. They identified six practices and associated improvement actions for each case. Furthermore, the practices were organised into an overall change improvement framework. The discussion pointed out that requirements changes during RE are different from later changes, for instance, with respect to contracts. It was stressed that over-constraining requirements changes might be just as dangerous as being too permissive. It was also indicated that a classification of various types of changes might be needed.

In the final talk of the session, Timo Käkölä (co-author: A.A. Salo) discussed requirements for groupware-support for RE processes which aim at developing new products. A list of main issues were identified and discussed based on experiences from large-scale development projects within Nokia. The ensuing discussion focused on the particularities of RE as an application area for CSCW and on which types of groupware tools that were most applicable. Asked to prioritise his list of issues, Käkölä responded that bridging the communication gap between people from marketing, R&D and production was particularly important in the projects studied. Also, the participants compared the Lotus-based approach taken at Nokia - based on replicated repositories - with less tightly integrated alternatives, e.g., based on viewpoints.

Discussion Summary

The session discussion focused on two topics:

- How do we recognise good practice when we see it?
- How do we characterise good practice when we have identified it?

Several ways of identifying good practice was suggested by the participants, including:

- company turnover,
- numbers of requirements-related defects in the final products,
- asking people who do RE well,
- the market value and stability of products,
- comparisons with RE theory,
- percentage of unnecessary requirements,
- number of process cycles needed for RE, and
- that the practice makes compliance checking possible.

In the discussion of characterisation, participants suggested that the process should:

- result in a well-decomposed problem space,
- produce clear goals,
- clarify costs and benefits of each alternative solution,
- incorporate rationales and arguments for why it is good practice,
- specify clear and applicable pre- and post-conditions for each activity step,
- transfer a great deal of domain-knowledge to developers,
- provide checklists to derive system characteristics,
- make users aware of the consequences of requirements, and
- adapt CMM-like maturity levels.

The session discussion then concluded with a discussion of whether compliance-checking is feasible for RE processes.

Session Summary: Position Papers and Preliminary Results 2

Lee Young (co-author: N. Mehandjiev) presented requirements engineering problems when developing management information systems, pointing to a knowledge, responsibility and communications gap between three groups of actors: strategic managers, tactical managers and systems analysts. The solution outlined was an exploration process for requirements decisions which centered around a multi-perspective model of strategic requirements. Lee also presented early experiences from an industrial case study.

In the second talk of the session, Joao Carvalho (co-author: I. Santos) presented a conceptual framework for studying how people understand the organisations they work in, the work they do, as well as their expectations for new software systems, challenging the common RE focus on technical issues. Joao pointed out that RE work has both structural, social, political and symbolic dimensions, and outlined how the framework could inspire an RE method. A research design for validating the framework was also outlined.

Session Summary: Specification and Validation

Overview of Talks

The session started with the presentation by Pär J. Ågerfalk (co-author: Göran Goldkuhl). Pär outlined a language action

approach for RE. He argued that it is vital to take the communicative aspects of information systems into account when talking about and defining the requirements of the systems. The main focus of the talk was on the perspectives under which the RE process is being performed. As main contribution he proposes to view the operationalisation of information systems as actions. In the discussion a more detailed comparison of the proposed approach with existing action oriented approaches, e.g., speech act approaches and business process approaches was requested. It was questioned if the resulting model is better suited to communicate with the stakeholders than existing models. In addition, a clearer statement of the achievements would help to understand and relate the proposed approach better.

Søren Lauesen (co-author: Houman Younessi) proposed six important facets to be considered when defining usability requirements for a system: the time it should take for an (experienced and inexperienced) user to perform the task (performance style); the rate for serious usability defects in performing a task detected by the user (defect style); the number of prototype cycles to be performed for building the product (process style); the satisfaction rate of users for learning and using the system (subjective style); the screen lookouts to be used by the system which are usability tested during RE (design style); the style guideline to be used, e.g. Windows have at most three levels. To define the usability requirements for a particular system function the six facets must be combined in an appropriated way. The approach has been successfully applied to real applications. In the paper related discussion the six facets were seen as a good guideline to improve the definition of usability requirements during RE. The guidelines provided could be improved by associated well-known HCI techniques to the facets to be used for eliciting the requirements.

The third talk in this session was given by Neil Maiden (co-authors: M. Cisse, H. Perez and D. Manual). Neil proposed a pattern language for socio-technical system design to inform the validation of system requirements. Patterns are used to relate scenarios and requirements. A pattern, e.g., an input of essential information to the system, is defined by a situation in which the pattern should be applied, and a set of scenarios by which the actual defined requirements can be tested against "common good" knowledge expressed in the scenarios. Using structured walkthroughs, not only the requirements are validated against the scenarios, but also reflection on the patterns itself is enabled, e.g., improving the scenarios, the situation of the patterns, is enabled. Neil sees patterns as an ideal means for managing organisation knowledge about good RE practice, i.e., to facilitate the reuse of experience in new projects. In the discussion, it was questioned if the use of patterns to organise and enable the reuse of experience is better than other experience based approaches used for several years. The approach should be applied to real projects to validate, among others, if the time needed to check the requirements against the patterns pays of and if patterns per se are well suited to organise, manage and improve organisational knowledge.

Discussion Summary

The general discussion focused on the general issue of industrial uptakes of research results. The absence of well-trained people in the area of RE in industry was seen as one factor that makes industrial uptake of research results complicated. Researchers should thus invest more time and effort in the training of people. Industrial uptakes, in general, takes place incrementally, i.e., it is quite unlikely that industrial uptake takes place in one big jump. Prerequisites to enable industrial uptake are the application of research results to real life cases, the integration with the methods used in industry, and the definition of easy-to-learn and easy-to-use guidelines (methods) which empower the application of the results in real projects. The discussion of some examples seems to indicate, that there is an increasing number of researchers which downsize their research results to make them industrially applicable. On the other hand, the results are often not promising enough, i.e., industry is most often not willing to apply (parts of) the results in trial applications. Nevertheless, the participants from industry and research agreed that closer co-operation are needed to improve RE practice, to provide feedback about research results, and to raise/enforce practice oriented research directions; lets get such co-operation started.

Session Summary: Industrial Problem Statements

Overview of Talks

To ensure that RE researchers react to the needs of practitioners REFSQ'98 included a special session of industrial problem statements. The session turned out to be a fruitful experience which will be continued at future workshops. The problem statements dealt with a diverse range of topics indicating an abundance of industrially relevant RE areas which are open for future research. The first statement, made by Sjaak Brinkkemper, discussed requirements management in the context of development of packaged software. The talk emphasised the need for academic RE research within an area characterised by many customers, many release cycles, platform independence and concurrent engineering of several aspects of the final products (software, documentation, training, etc.). Particular research issues suggested by Sjaak are: bundling of requirements, impact analysis after requirements changes, estimates of changed resource needs, interference between requirements and problem tracking, managing relationships between requirements, the separation of specific from standard requirements, specification of requirements by non-technical persons, as well as requirement management tools.

In his statement, Paris Bayias (co-authors: T. Hadzilacos and P. Velonias) presented problems with applying existing RE methods when the new system is heavily constrained by legacy systems. The talk reported experience from the OASIS automated electronic stock exchange system and outlined the RE approach used to establish requirements for OASIS. In the discussion, it was pointed out that the project seemed to have particular timeliness and stability requirements. It was also pointed out that the lack of an

underlying representation framework might represent a problem. Dieter Landes emphasised the importance of non-functional requirements based on experience from applying the Goal-Question-Metric paradigm to several RE projects at Daimler-Benz. Dieter stressed the importance of tightly coupling and assessing non-functional requirements in parallel with functional ones. The discussion pointed to the need for cost assessments in association with a methodology. However, since cost tends to be fixed, the practical consequence is more often a need to bundle and prioritise among the derived requirements. The usefulness of distinguishing between functional and non-functional requirements was also briefly discussed.

Finally, Marcelo Masera (co-author: P. Morris) reported the results from two workshops organised by the EC's Joint Research Centre. The workshops aimed at identifying industrial RE research needs related to development of embedded systems. The extent to which existing industrial RE approaches and standards - mainly developed by and for large organisations with "high-end" RE needs - would scale to the majority of enterprises was controversially discussed.

Session Summary: Scenarios in RE

Overview of Talks

The last paper session of the workshop focused on the use of scenarios in RE. It continued along the session on scenarios and use cases of last year's REFSQ workshop.

Björn Regnell (co-author: Per Runeson) presented a number of possible combinations of scenario-based RE with static verification and dynamic testing. He especially focused on the opportunity of using prioritised scenarios for guiding inspections and the combination of scenarios with reliability testing. The ability to derive test cases from scenarios clearly depends on the representation formalism used. Which scenario presentation is better for which type of test cases has to be evaluated. Björn stated the hypotheses that usage-based reading using prioritised scenarios will elicit the defects which matters most, that reliability tests are less difficult and less expensive to construct if they are based on scenarios, and that EP-analysis gives a more complete set of scenarios compared to ad-hoc scenario elicitation. Future work will test these hypotheses by defining concrete scenario-based test approaches and applying them to real case studies. In the discussion, the need for relating scenarios and test cases was controversially discussed. On the one hand, test generation approaches could be much more productive. On the other hand, proving to the customer that the developed software can deal with certain scenarios (use situations) in the defined way can better be achieved with scenario-based test approaches. It is an open research issue for which test situations (purposes) which approach should be applied.

The second talk of the session by Barbara Paech dealt with extensions to use cases to empower richer and thus better use case description. Barbara identified four levels of information to be included in a use case description, namely work division together with application data, application-specific system functions, work-specific system functions and dialogue definitions. The goal of extending use cases along the lines suggested, is to improve the fitness for use by integrating HCI, RE, conceptual modelling and OO development ideas. The proposed extensions should allow to express and interrelate different views on "use cases". In the discussion, guidance on how to define those level was requested. Moreover, the benefits gained by defining use cases using this more complex notation should be validated by applying the approach to real life problems. Thereby it could be tested if the four levels are sufficient and if they improve the use case usage.

In the last presentation of the workshop, Shailey Minocha (co-author: Alistair Sutcliffe) presented an approach for scenario-based analysis of non-functional requirements. The main goal of the work is the guided elicitation of non functional requirements (NFRs) using scenario templates. Based on an extensive literature survey and existing classification frameworks for NFRs, an comprehensive list of interrelations between NFRs and quality criteria was defined. Based on the relation to the quality criteria the interference of different NFRs was elaborated. For seven NFRs (usability, reliability, reusability, portability, security, scalability, performance efficiency) templates were proposed which aggregate advice on process and representation for analysing the NFRs in concrete projects. Those templates were then embedded in an NFR analysis method. The approach was seen as very ambitious, since each NFR is a major area by its own and thus requires a major effort to deal with it successfully. Thus, a restriction to a more narrow theme, i.e. to one NFR (e.g., security requirements) and an in-depth elaboration with domain experts was suggested. It was questioned if templates (patterns) really help to identify and specific NFRs.

Discussion Summary

The session discussion took the form of a brainstorm on open and/or problematic issues in relation to scenarios. The following issues were mentioned:

- How to ensure that the chosen scenarios have the right coverage?
- How to make cost/benefit-analyses with respect to scenarios? Both measures and empirical studies are needed.
- How to integrate scenarios with other RE and development artefacts; requirements in particular?
- How to structure the documentation of scenarios for reuse?
- How to relate scenarios to change management and impact analysis?
- How to prioritise scenarios and how to use them for prioritising requirements?
- How to manage evolving scenarios and ensure responsiveness to change - in particular for scenarios at the strategic level?

- How to assess the quality of scenarios? (Defects, inspection, etc.)
- How to classify scenarios? One classification has been suggested by the CREWS project.

General Discussion

The starting point of the general discussion was the need to validate RE research in industrial settings. Although it was pointed out that REFSQ has clearly evolved in this direction over the years, it should be a goal of next year's REFSQ to have even more presentations that reflect experience from practical examples.

It was pointed out that it is difficult to make commercial companies use new and yet unproven methods, techniques and tools. Hence the situation calls for new methodological approaches to validate research results.

Industrial participants pointed out that the scarce resource for industry is human time due to the scarcity of domain knowledge, not money! From the academic side it was stated that it is difficult to get clear problem statements from industry. The available surveys on states-of-practice are mostly done by academics.

As a preparation for REFSQ'99, a list of RE problem areas which had been collected throughout the workshop was presented and discussed. The list especially emphasised industrial problems and needs:

- bundling requirements into COTS releases,
- analysing the impact of requirements changes,
- resource estimation,
- establishing and using traceability,
- configuration management for requirements,
- support for change management,
- classification of COTS requirements into domain-specific and standard ones,
- consideration of existing systems,
- consideration of quality requirements,
- smoothly extending existing methods,
- assess the helpfulness of formal representations,
- develop techniques that are specific for individual domains and applications, and
- consider techniques that are specific for COTS.

Andreas L. Opdahl and Klaus Pohl
Pisa, Italy, June 1998

References

[1] Eric Dubios, Andreas Opdahl, Klaus Pohl, "Workshop Summaries REFSQ '97"; in: Proceedings of the REFSQ '97 Workshop, F.U.N.D.P, Namur, Belgium, 1997

[2] Klaus Pohl; "Process-Centered Requirements Engineering"; RSP/Wiley ands Sons, UK, 1996

CORE - Blimey!

A regular column by Geoff Mullery, of Systematic Methods Ltd.

YASB - What's in a Word?

In recent contributions I have tried to explain prerequisites for a comprehensive support environment and some inadequacies in existing technical tools and techniques. In this contribution I shall continue the theme, but concentrating on more general purpose support tools.

I start with a problem of interpretation based on visual cues in an environment where people will either impose alternative interpretations on the cues or ignore them and interpret only the words. First, let me introduce four sentences pairs involving the same words, but with differing visual cues:

The dog sat on the rug, the cat sat on the mat.
The dog sat on the rug, the ***cat*** sat on the mat.
The dog sat on the rug, the cat sat on the ***mat***.
The dog sat on the ***rug***, the ***cat*** sat on the mat.

These sentence pairs, three of which make use of visual cues in the form of bold/italic print formats, invite different interpretations of exactly the same words. The first is simply a list of animal/object associations – {dog, rug} and {cat, mat}. The second corrects a misinterpretation of which animal sat on the mat. The third corrects a misinterpretation of which object the cat sat on. The fourth corrects an incorrect switch of which animal sat on which object.

With these interpretations neither the first nor the fourth pair implies the need for another animal or object but the second implies the existence of another animal, the third implies the existence of another object. The fourth pair implies a prior, incorrect juxtaposition of the dog and cat.

What I am showing here is that visual cues contain or can be interpreted as containing key information in a specification. The problem with visual cues is that the key information is implied, not explicit. In fact the second and third pairs could be given the interpretation I gave to the fourth pair. I led you into alternative interpretations of them. I may have been incorrect to do so.

This problem is not new. It is one reason advocates of formal methods are so keen on a formal definition which clarifies both the syntax and semantics of any notation used for a contractually binding specification. As the richness of the notation increases (e.g. I could add underlining, font type and size variations, layout rules, diagrams, pictures and document frames) the difficulty of achieving a correct interpretation increases dramatically without a formal definition of the notation.

I have mentioned in earlier contributions the problem of getting people to assist in translation between informal and

formal notations. Failure to allow for such translations in cases where they are vital is at the root of many problems of major system development failures. The larger, more complex, more expensive, more long-lived, more safety-critical (in terms of people's physical, mental and financial well-being) a system is the more vital it is to have a formal statement of what is to be and what has been implemented.

Often the problem is not that one approach (formal or informal) has been used instead of the other, but rather that only one of the approaches has been used. I have worked a lot in military system environments and somewhat less in air traffic control system environments and in both cases I have seen various levels of use of formal or informal approaches, but they all founder.

In passing I must say that the primary reason for most failures in development of large systems is not purely technical. It is managerial and contractual, but the reasons such failures occur are basically the same as for technical failures – people make informal, contradictory and multiply interpretable statements and the contradictions and interpretations are not satisfactorily resolved, or are resolved too late to prevent major waste of time or money, leading to delivery of inadequate and sometimes dangerous systems.

The root problem is failure to support derivation, linkage and propagation of information using forms of expression ranging from highly informal through to strongly formal and failure to support the expression of conflicts, their tracking and resolution. A comprehensive support environment must be capable of supporting any and all these facilities without making any always mandatory.

Avoiding a universally mandatory approach is important because there are system developments which do not require the formality necessary in things like military and air traffic control systems. As a technician you may wish to get as near to a perfect product as you can, but if you look at other industries with a reasonable track record in customer satisfaction you will see they produce a range of product quality options, usually requiring greater payment for greater quality.

When achievement of quality is almost wholly automated we can consider producing a single (high) quality product option. When so much requires subjective judgement, as is the case with major computer systems at present, then a range of quality options is necessary and thence a comprehensive support environment must support such a range.

So how is all this reflected in deficiencies in current tools? Well, it is basically related to their inability to support association and propagation of informal statements and formal interpretations derived from them by other interested parties and tools. It is also true that there is little support for association and propagation of formal statements between different technical tools, apart from a few in a bespoke tool set such as a programmer's development environment.

This failure takes two related forms. First there is failure to provide a sensible way of ensuring that those with an interest in a topic will find out about the existence of a new publication which addresses it. Then there is failure to provide a way of associating interpretations of parts of a publication with the source material and to propagate those interpretations to all interested parties.

There are three types of tool approach which partially address these needs: Web-linking, Search Engines and Post-it Note add-ins. They are individually and collectively inadequate and I have not seen much sign that they are moving in a direction which will redress this failing.

Web-linking is by consensus a big improvement on what is basically the linear form imposed by paper documents. It allows the author of a publication to identify key linkages from one place in a document to another with a related topic. Unfortunately that facility is restricted to the author of a document, which precludes a reader from defining any more than a fairly gross linkage between his/her interpretations and the original information to which it relates.

In particular, an interpretation needs to relate to not just one point in a document it interprets, but potentially to a chain of related points. For example, to resolve the interpretation of the second and third of the animal/object association sentences above I would need to point to that sentence and to the context which established the original animal/object association it was qualifying – otherwise my interpretation could not safely be judged for correctness.

Remember also that there may be competing interpretations by others and thence there may be a need for a conflict-resolving ruling interpretation, which may be made by the writer of the original document or by a manager delegated decision-making authority. This illustrates that the support environment must permit the presence and resolution of conflict and ambiguity.

Search engines are notorious for their crudeness in establishing a true match between their users' true search requirements and the available documents. Though there is some movement towards more sophisticated search techniques there remains the problem that I may enter my search today and find nothing, but someone may publish a relevant document tomorrow or next week or next month and I will find out about it only if I re-enter my search repeatedly.

In reality I need to define my search, together with parameters such as for how long I want it to remain current and over what range of destinations I wish to search (e.g. am I looking for a UK source, a European or US source? Am I looking for a commercial or university source or a research institute source? Am I looking in places where the search will cost me money or only where it is free – and if I am prepared to pay for the search, what is the upper limit on cost? Are there places I wish to avoid searching, such as rival organisations?).

Post-it Note add-ins to word processing packages allow for comments to be made and read, but the comments are hidden from anyone other than readers of the document. There are certainly cases where that is all the originator intends, but where information availability and flow are critical the fact that the originator does not think a comment has wider applicability does not mean that that is so.

I have worked in many project situations where the author of a document or of a review/commentary do not realise there are other people on the same or related projects desperately seeking information on the same topic. The bigger the project the wider the scope for this problem. It is endemic in military systems projects I have worked on.

For these problems to be reduced I believe there needs to be a re-think about the purpose and scope of Web-linking languages like HTML and of the use of Post-it note facilities. There also needs to be a significant expansion in the capabilities and mode of operation of search engines.

In another context I have co-authored a crude definition of a facility called an Enabling Network System (ENS) to deal with some of these needs and I will explain a little more about that in a later contribution. I can see no barrier to implementing an ENS in an Intranet/Internet environment as it evolves - but it does require further evolution and I the appropriate evolution is not currently evident, though the components for its occurrence seem to be in place.

In conclusion, for those of you who have seen me doing hatchet jobs on systems "solutions" presented at conferences and seminars, what I have said in this contribution is a major factor in my destructiveness. I have never seen a public presentation of an approach which was entirely without merit, but I regularly see presentations which imply they have the one true way to development success. They only back off and hedge their bets when I, or someone else wields the hatchet.

If you have a support environment and it can't support all the things I have mentioned here across a range of tools rather than your own bespoke tool set, or if it can't support some of the other features of method and tool I have mentioned in previous contributions, then it is not a comprehensive solution. If you present it as anything other than a partial solution then if I am in the audience, I will draw the hatchet and accuse you of having YASB – Yet Another Silver Bullet.

Geoff Mullery
geoff_mullery@dial.pipex.com

RE-Publications

Book Review "Requirements Engineering – Processes and Techniques" by Gerald Kotonya and Ian Sommerville. John Wiley 1998. ISBN 0-471-97208-8.

Reviewed by Ian Alexander

This book is the companion to Requirements Engineering: A Good Practice Guide (Ian Sommerville and Pete Sawyer, John Wiley 1997) (reviewed in RQ 12, October 1997). There is as the authors say very little overlap: the *Guide* gave short bits of advice whereas *Processes and Techniques* is a systematic textbook.

The book is aimed at trainee software engineers at second year undergraduate level and above. It is very plainly and clearly written without jargon or pomposity, and it manages to be fair to all sides in the structured/object-oriented wars. It accordingly covers both traditional and modern industrial approaches such as SADT and CORE.

The book is structured into two parts: 5 chapters on the RE process (the 'what'), and 5 on RE techniques (the 'how').

Part 1 contains an excellent introduction which includes FAQs about RE, its place within system engineering, and the nature of the requirements document. There follow chapters on processes, elicitation, validation, and requirements management.

Part 2 introduces the various methods for RE, including data-flow modelling, semantic data models, object-oriented approaches, and formal methods. There follow chapters on viewpoint-oriented requirements methods including SADT, CORE, VOSE, and VORD, non-functional requirements, and interactive system specification. The book closes with a case study for a (real) electronic document distribution system which the authors were involved in.

Each chapter is well and helpfully organized, aided by a clean and spacious page design. Each chapter begins with a short Contents list and a one-paragraph summary. The body of the chapter is a discussion of the topic, divided into sections and often with numbered or bulleted points for the student to distinguish. The text is well supported by line diagrams and tables, which like the text also focus on classifying the various factors involved. The chapters close with a list of key points, a set of five or ten exercises, a list of references, and some useful suggestions for further reading.

There is a short but effective index. One topic unaccountably lacking both from the index and the book is verification / testing; validation is mentioned briefly in the introduction but there is no discussion of, for instance, the use of

requirements to generate test scripts, or the importance of tracing between requirements and system tests.

The book was written before Use Cases emerged as a fashionable topic, though they are mentioned in the discussion of Jacobson's object(ory) method. Rumbaugh's OMT and scenarios also get brief mentions. On UML, the book merely observes that

> "efforts are underway to integrate several object-oriented methods to create a tailorable universal method, and these are likely to come to fruition in the next few years" !

The authors, in a similarly wry tone, state that several object-oriented methods have been proposed, from Colbert, Coad & Yourdon, Rumbaugh, Jacobson, and others, and that

> "These notations are semantically similar and any of them could be used for model description".

The section on methods wisely concentrates on the principles, supporting the discussion with examples worked through in sufficient depth to make clear how the methods work.

The authors have chosen to avoid explicit mention of requirements tools, though there are sensible discussions of related topics, such as of techniques for requirements identification and traceability, leading to the conclusion that you need a database "with each requirement represented as one or more database entities. The facilities of the database can be used to link related requirements…" The discussion compares relational and object-oriented approaches, concluding that the latter "are structurally more suited to requirements management". The only tools actually named are ORACLE and INGRES, however:

> "Commercial requirements management systems usually rely on such commercial databases for requirements storage."

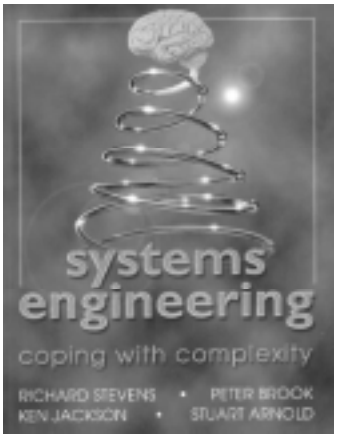
This claim is hard to quantify, but is untrue both of those tools which rely on cheap PC databases, and of those which include a specialised requirements database engine. The practical advantages of using a tool designed for managing requirements are perhaps slightly overlooked.

Requirements Engineering – Processes and Techniques comes with its own website – http://www.comp.lancs.ac.uk/computing/resources/re so you can find out more and keep in touch with updates to the authors' thinking.

The book is a solid and useful reference on the more established current techniques for practitioners, an excellent up-to-date guide to the literature, and a practical university text. The careful blend of theory, pragmatism and worked examples is especially welcome. As such, this is one of the first textbooks on RE that can be recommended almost unreservedly to RQ readers.

Book Review "Systems Engineering – coping with complexity" by Richard Stevens, Peter Brook, Ken Jackson and Stuart Arnold, Prentice Hall, 1998

Reviewed by Ian Alexander



In contrast to Kotonya's *Requirements Engineering* (reviewed above), Stevens' *Systems Engineering* looks at the place of requirements in a world which consists of complex systems in a highly competitive marketplace. This may be the commercial world or equally the military-industrial world in which systems must literally do battle with their rivals.

Stevens and his co-authors (two of them from the UK's Defence Evaluation and Research Agency) know that in this environment, many systems fail, very often because they were inadequately thought out, and often also because their development projects were poorly managed. Chapter 1 begins "The world is currently gripped by changes more intense and rapid than those triggered by the industrial revolution…" : we are at once swept into the rich, complex, and dangerous life of real system development.

For Stevens, the problem in systems engineering is complexity, and its mastery is, as the subtitle implies, the key to success. The design of complex systems demands hierarchy – of organisations, of projects, of contracts, of documents. Hierarchy implies interfaces: if you split a system into three, you probably create three interfaces between the component subsystems. Interfaces in turn imply specialisation: someone develops the hardware; someone else, the software. Similarly, someone (the customer) writes the requirements specification, while someone else (the developer) tries to meet those requirements. This, like the prime contractor – subcontractor relationship, consists of a customer and a supplier: the marketplace reaches right into the core of system engineering.

The book therefore covers a startling breadth of subjects, but always with the same practical vision and with the same conceptual tools. The first few chapters broadly follow the European Space Agency's now-classical PSS-05 software engineering standard life-cycle phases: user requirements, system requirements, architectural design, integration (of subsystems) and verification, management. (Requirements are involved in every one of these phases.) Once the reader is grounded in the basics, the next chapter discusses how to tailor the simple life-cycle just presented. A tell-tale section entitled 'smaller systems' gives the game away: the systems

in the authors' minds are a great deal larger than the PC 'systems' beloved of advertising copywriters.

The second part of the book (chapter 8 onwards) starts by looking at more realistic life-cycles, based on the management of risk: when is it sensible to go ahead with something? The answer is, when success can be assured even if the bad risks materialize. That can only be guaranteed if the risks have been quantified. Concepts of requirement priority and benefit, risk, and cost loom much larger in the marketplace than technical issues.

The remaining chapters examine management in multi-level projects (hierarchy again), software and systems, prototyping (to control risk), information modeling, projects and the enterprise, a chapter on how to improve and a summary.

Each chapter consists of a double-page title/table of contents, overlaid on some crisp pencil artwork on the theme of engineering progress (from Leonardo's hang-glider to an agile jet). The text is broken up by plenty of simple flow diagrams illustrating life-cycles, trade-offs, business processes and information models, as well as short summaries of what the most important system documents should contain. Key points are highlighted or bulleted within the text. The chapters end with a page or two of realistically tricky exercises: the answers cannot be coded in C.

The helpful appendices include a list of websites: *Systems Engineering* comes with its own website – http://www.complexsystems.com which contains pointers to several related sites, and itself includes 'proposed' answers to the exercises which end each chapter. Students will find the glossary helpful and comprehensive. There is an extensive list of very varied references, and a detailed index.

This book is a carefully worked out description of the process of developing any large, complex, and risky system. It can also be read as a polemic: an impassioned plea for the discipline to graduate from its narrow roots, whether in academia or in quality control. The concluding paragraphs make it clear that system engineering is a human process, a 'game' in which there are losers as well as winners, something that can be played well, and that absolutely must be played better to limit the risks and losses that are still all too common.

RQ readers may feel like challenging some of the propositions in this thought-provoking account. Is it necessarily true, for instance, that firms must compete in a marketplace: what will happen if we end up with only one major aircraft manufacturer in the world, say? And is a Darwinian-capitalist worldview the only option? If it melted away like its old cold-war rival, what could we create to replace it? Whatever the reader's position, *Systems Engineering* is a powerfully stated argument, which fills a major gap in the RE literature: all future texts will have it in mind.

The role of requirements engineering within the industrial context is especially well expounded, and from a point of view that is both practical and quite unlike any of the other RE texts that have been reviewed in issues of RQ. The book will be of interest to several quite different communities: in particular development managers, clients having large systems developed, and students of system and software engineering will all find much that is of interest here. The book may also be a useful supplement (or perhaps an antidote) to the academic perspective on RE. All RQ readers should have access to a copy.

Note: The Malvern sub branch of the BCS (in conjunction with INCOSE) will be holding seminar at DERA Malvern, 2-5pm, Thurs 8 Oct, on ***Systems and Software Engineering*** *We expect the four authors will be there, and there will be a mixture of presentations and discussions.*

RE-Bites...

Requirements conflicts can be hard to detect. Part of the problem is that the questions of whether any two requirements conflict may be entirely dependent upon context.

Take for example the California highway laws (which can be regarded as requirements for drivers). Obviously, there is a speed limit, which for most highways is 65mph. There is also a law that requires drivers to keep up with the other traffic on the road. No conflict yet, right? However, what happens if you enter a highway where all the other cars are doing 85? (This is actually quite typical in California – there's no notion of a fast lane and a slow lane).

So, context is important. There may be requirements that don't appear to be in conflict at all, until some unusual context is encountered, and identifying these contexts ahead of time is hard.

RE-Calls

Recent Calls for Papers

Fourth IEEE International Symposium On Requirements Engineering (RE'99), University of Limerick, Ireland, 7-11 June 1999

THEME: Requirements in a Changing World

http://www.ul.ie/~isre99

Sponsored by the IEEE Computer Society (pending) in cooperation with ACM SIGSOFT (pending) and IFIP Working Group 2.9 (Software Requirements Engineering) (pending) and RENOIR (Network of Excellence in Requirements Engineering)

Requirements engineering (RE) involves the gathering, documenting, validating and use of requirements for software-intensive systems from all stakeholders who may be affected by the resulting system. As software products penetrate further and increasingly invisibly into everyone's personal and professional lives, it is becoming more vital that these products be developed to meet real requirements and that RE become part of every development organization's development process. RE includes all the technical, managerial, and interpersonal techniques involved in meeting this goal.

RE is emerging as a mature field of research and practice within software engineering and information systems, and has widespread implications in such allied fields as human-computer interaction, organizational design and systems engineering. All these fields deal with technological and procedural interventions into some existing states of affairs: a manual operation is automated, an existing software system is enhanced, a work practice is augmented with tools, an engineering system is monitored and controlled by software. Requirements for new or enhanced systems, therefore, come from changes in the world: changes in people's expectations, in business practices, in social forces, in enabling technology opportunities.

Examples of the changes that may be discussed at RE99 include:

- the transition from custom systems development to the integration of off-the-shelf products and online services;
- the growing concern with end-user responsiveness and reduced development cycle times;
- the ubiquitous embedding of software artifacts in the industrial environments and in the everyday activities of business and home life;
- the restructuring of the business world so that an enterprise is increasingly dependent on others to meet its own requirements.

The RE series of international symposia has established itself as a prestigious gathering of requirements engineering researchers and practitioners. RE99, the fourth in the series, will be held on the scenic campus of University of Limerick, Ireland. It will bring together researchers and practitioners of requirements engineering to discuss and exchange the latest developments and concepts. The program will consist of invited talks, paper presentations, panels, tutorials, birds of a feather sessions, demonstrations, and a doctoral consortium.

Contributions

Papers

Submissions on all aspects of requirements engineering are invited. Contributions with direct relevance to the conference theme are especially welcome. We will encourage interaction between practitioners with a commitment to step back and reflect on their knowledge and members of the research community who have a commitment to the practical evaluation of their ideas.

All submissions will be subject to critical review and will be expected to advance scholarship in the area of RE in one of several ways. Below are the categories of submission and their evaluation criteria.

Discovery and invention: A new language, method or automated way of working that supports the RE process. Contributions should be novel. They need not be empirically validated, but should include a serious discussion of their relevance and practicality.

Integration and analysis: An integration, review or analysis of previously known ideas that presents them in a fresh light. The ideas that are integrated do not have to be novel, but the integration must be insightful and useful to those already familiar with them.

Application and evaluation: One or more cases of a new or previously described technique being used in practice. Qualitative and metrics-based evaluations are equally welcome, but in all cases there should be serious reflection that goes beyond the case or cases at hand. Contributions should be sound and potentially useful to practitioners.

Scholarship of education and learning: Descriptions of courses in requirements engineering or the treatment of RE in other courses together with rationale and evaluation. Contributions may deal with higher education or professional courses. There should be a discussion of the implications to the future of the profession.

Within these categories, papers are welcome in any area of requirements engineering. Symposium organizers extend a special invitation for paper submission and participation by researchers and practitioners working in areas directly relevant to the conference theme including, but not limited to:

- Agent oriented RE, COTS - component based RE Legacy systems
- Global RE - doing RE on the Internet
- Ubiquity and Interoperability - new demands on RE

Panels

Proposals for panel sessions should have one topical question to be discussed or debated by the panelists. Novel

strategies for managing the panel and stimulating audience participation are especially encouraged.

Doctoral Student Symposium

A one-day symposium is being arranged for graduate (post-graduate) students who have made substantial progress in their RE work but are not expecting to graduate for at least one year. Students will present their work-in-progress to selected senior members of the Requirements Engineering community, for purposes of feedback and discussion.

Presenters will be selected on the basis of a research summary, not exceeding 2,000 words in length. The reports of presenters will be published as notes for consortium participants. Limited financial assistance will be available for presenters. Doctoral students should contact the consortium chair for details.

Demonstrations

Automated tools may be demonstrated during sessions that will be scheduled in parallel with the presentations and panel sessions. Technologies developed by not-for-profit and commercial organizations are welcome. However, commercial demonstrations should be given by development or technical support staff who are able to provide detailed information about the tools demonstrated.

The symposium will provide free facilities for demonstration of selected tools. Potential demonstrators should submit a description of the system they propose to demonstrate, not to exceed 1000 words, to the general chair.

Tutorials

One or two days before the technical program will be devoted to half-day and full-day tutorials. A tutorial submission should emphasize the audience for whom the tutorial is intended and their learning objectives.

Information For Authors

Authors must submit each submission using the WWW based facility available through the RE'99 World Wide Web site, http://www.ul.ie/~isre99. Each submission must be in PDF (preferred) or plain Postscript format.

Submissions should be unpublished and not under consideration by another conference or journal. Papers should not exceed 6,000 words in length. Each paper should be accompanied by full contact information including name, address, E-mail address, telephone number, and fax number.

To aid the PC in planning the conference, authors are also requested to submit an ASCII text copy of the title and abstract of each paper using the WWW based facility available at the RE'99 World Wide Web site, http://www.ul.ie/~isre99, two weeks before the full paper is due.

All accepted papers will appear in the proceedings of the symposium, to be published by the IEEE Computer Society Press. The best papers will also be considered for publication in a special issue of the Requirements Engineering Journal.

Important Dates

2 Nov 1998, paper title, abstract received by RE'99 via WWW submission.
23 Nov 1998, full papers, panel proposals and tutorial proposals due.
1 Feb 1999, notification of acceptance of accepted papers, panel proposals, and tutorial proposals; deadline for doctoral student symposium submissions and demonstration proposals.
1 March 1999, camera-ready copy of accepted papers and panel introductions due; notification of acceptance of doctoral student symposium submissions and demonstration proposals; receipt of audience handout master copies for accepted tutorials.

RE99 Organizing Committee

General Chair

Prof. Kevin Ryan
University of Limerick, Ireland
Kevin.Ryan@ul.ie

Program Chair

Prof. William N. Robinson
Georgia State University, USA
wrobinson@gsu.edu

CAiSE 99 : 11th International Conference On Advanced Information Systems Engineering, Heidelberg, Germany, June 14-18, 1999

(http://www-i5.informatik.rwth-aachen.de/caise99)

Important Deadlines

Oct. 30, 1998: workshop proposals
Nov. 30, 1998: paper submissions, panel & tutorial proposals

Conference Aims & Objectives

Since the late 1980's, the CAiSE conferences provide a forum for presentation and exchange of research results and practical experiences within the field of Information Systems Engineering. CAiSE*99 will be held in Heidelberg, not only a beautiful town with the oldest university in Germany, but also close to numerous leading Information Systems companies in the Rhein-Main-Neckar area around Frankfurt. In tune with this environment, a major theme of CAiSE*99 will be 'component-oriented information systems engineering'.

CAiSE*99 aims at bringing together researchers, users, and practitioners in the field. The conference programme will feature paper presentations, workshops, tutorials, and interactive panel sessions.

CAiSE*99 is organized in cooperation with the Information Systems Engineering and Business Information Systems groups of GI (the German Informatics Society), and with AIS, the International Association for Information Systems.

Conference Theme

Component-based approaches mark the maturity of any engineering discipline. However, transfering this idea to the complex and diverse world of information systems engineering has proven more difficult than expected. We seem to be drowning in a flood of poorly organized information rather than moving towards an organized component-based world. Despite numerous proposals from object-oriented programming, design patterns and frameworks, customizable reference models and standard software, requirements engineering and business re-engineering, web-based systems, data reduction strategies, knowledge management and modularized education, the question how to make component-oriented approaches really work in information systems remains wide open. We are seeking papers and tutorial proposals towards answering these questions from a wide range of perspectives but with the common goal of making the theory and practice of component-based construction and evolution of information systems work.

Relevant Topics

In addition to the special theme, general topics relevant for submissions to CAiSE*99 include, but are not limited to, the following:

- Methodologies and Models for IS,
- Interrelating Methods and Formalisms,
- Reuse at Conceptual and Design Levels,
- IS Implementation Techniques, Languages,
- Application Generators,
- Object-oriented Design, Process Models,
- Interoperability of Applications,
- Spatial and Temporal Information Management,
- IS Strategy and Planning,
- Systems Procurement Processes,
- Maintenance and Evolution of IS,
- Information Repositories,
- Data and Process Re-engineering,
- Metrics and Indicators,
- IS Security,
- Re-Engineering Legacy IS,
- Managing Organizational Knowledge Assets,
- Workflow Management and Co-operative Work,
- Case Studies and Experience Reports,
- Web Applications,
- Network-centric Computing,
- Information Services on the Net,
- Novel IS Architectures (Distributed; Co-operative; Intelligent; Interoperable; Multimedia; Open),
- Data Warehouses and Data Mining,
- Information Systems and Cultural Heritage.

Submission Information

All submissions, papers as well as proposals for workshops, tutorials, panels and posters, must arrive by the deadlines with the CAiSE*99 Program Chair.

Call For Papers

Five (5) printed copies of paper submissions should be sent to the program chair. The paper should not exceed 5,000 words. Papers must be original, and not submitted to, or accepted by, any conference or journal. A separate cover sheet should be enclosed containing the title of the paper, the author(s) and affiliation(s), and the address (including e-mail address and fax number) to which correspondence should be sent.

Proceedings from CAiSE*99 will be published in the Springer Lecture Notes in Computer Science. The proceedings for the CAiSE annual conferences from 1990 to 1998 are also available from this series.

The best papers of the conference will be proposed for publication (after revision and additional refereeing) in a special issue of Information Systems: The International Journal.

Call For Panels And Tutorials

Proposals are solicited for tutorials that will be held together with the conference. Tutorials can be half day (3 hours) or full day (6 hours) and should be of interest to participants from the academic and business communities. The conference programme will also include a few panels on current topics about which there is controversy and thus a need for public discussion.

Panel and tutorial proposals should include a title, an abstract of the topics, duration, and a copy of the proposer(s) curriculum vitae. Tutorial proposals should also clearly identify the intended audience and the requested level of expertise for attending the tutorial.

Call For Workshops

In conjunction with CAiSE*99, a series of workshops will take place. They are meant to facilitate the exchange of ideas and experiences between active researchers. Workshop topics will be in line with the conference topics. We invite prospective workshop organisers to submit proposals stating the topic and goals of the workshop. Proposals will be reviewed by the Organising Committee. All participants are expected to attend the main conference.

Call For Posters

During the CAiSE*99 conference, a special room will be reserved for poster sessions. Industrial and academic research projects of any scale are invited to illustrate innovative concepts, theories or prototype systems on a poster. Poster proposals should include title, one-page outline and names of presenters.

Important Dates

Deadlines:

Oct. 30, 1998: workshop proposals
Nov. 30, 1998: paper submissions, panel & tutorial proposals
March 1, 1999: poster descriptions

Notifications:

Dec. 1, 1998: workshop acceptance
Feb. 1, 1999: paper, panel and tutorial acceptance
April 1, 1999: poster acceptance

Workshop dates: June 14-15, 1999

Conference dates: June 16-18, 1999

Conference Organisation

Advisory Committee

Janis Bubenko
Royal Institute of Technology, Sweden

Arne Solvberg
University of Trondheim, Norway

Program Chair

Matthias Jarke
CAiSE 99
RWTH Aachen, Informatik V
Ahornstr. 55
D-52074 Aachen, Germany
e-mail: caise99@i5.informatik.rwth-aachen.de
phone: +49 241 80 21 501
fax: +49 241 8888 148

Organising Chair

Andreas Oberweis
J.W. Goethe-Universitaet, Wirtschaftsinformatik II
Merton Str. 17
D-60325 Frankfurt/Main, Germany
e-mail: oberweis@wiwi.uni-frankfurt.de
phone: +49 69 798-28722/28998
fax: +49 69 798-25073

Tutorial Chair

Klaus Pohl
RWTH Aachen, Informatik V
email: pohl@informatik.rwth-aachen.de

Workshop and Poster Chair

Gottfried Vossen
Universitaet Muenster, Wirtschaftsinformatik
email: vossen@helios.uni-muenster.de

RE-Sources

For a full listing of books, mailing lists, web pages and tools that have appeared in this section in previous newsletters, see the RQ archive: http://research.ivv.nasa.gov/~steve/resg/

Web Pages

The BCS RESG home page can be found at:
http://porta.cs.york.ac.uk/bcs/resg/

Back issues of Requirenautics Quarterly:
http://research.ivv.nasa.gov/~steve/resg/

Compendium of Software Engineering Tools:
http://www.methods-tools.com

Books

G Kotonya and I Sommerville, “Requirements Engineering” John Wiley & Sons, ISBN 0471972088, 1998

I Sommerville and P Sawyer “Requirements Engineering: A Good Practice Guide”, John Wiley & Sons, ISBN 0471974447, 1997

Technical Reports

CREWS Report Series

The CREWS project (Cooperative Requirements Engineering With Scenarios) is a three year (August 1996 - July 1999) ESPRIT Reactive Long Term Research project (21.903) from RWTH Aachen (Coordinator), City University, University of Paris I, and FUNDP-University of Namur. The project develops, evaluates, and demonstrates the applicability of methods and tools for cooperative scenario-based requirements elicitation and validation.

Abstracts and reports are available from

- ftp://ftp.informatik.rwth-aachen.de/pub/CREWS
- or
- http://SUNSITE.informatik.rwth-aachen.de/CREWS/

The reports are stored in compressed PostScript-Format using gzip. To get a specific report, transfer the desired file in binary mode and uncompress it.

Please, send comments and questions on the report series to crewsrep@i5.informatik.rwth-aachen.de

Mailing lists

Software Requirements Engineering Mailing List

To subscribe to the Software Requirements Engineering (SRE) mailing list, e-mail listproc@jrcase.mq.edu.au, with the only line in the body of the message:
subscribe SRE your-first-name your-second-name

Articles to the SRE mailing list should be sent to

SRE@jrcase.mq.edu.au.

RE-Actors

The committee of RESG

Chair: Dr. Bashar Nuseibeh, Department of Computing, Imperial College, 180 Queen's Gate, London SW7 2BZ. ban@doc.ic.ac.uk, Tel: 0171-594-8286, Fax: 0171-581-8024

Treasurer: Dr. Neil Maiden, Centre for HCI Design, City University, Northampton Square, London EC1V OHB. N.A.M.Maiden@city.ac.uk, Tel: 0171-477-8412, Fax: 0171-477-8859

Secretary: Wolfgang Emmerich, City University, Department of Computer Science, Northampton Square, London EC1V OHB. W.Emmerich@cs.ucl.ac.uk, Tel: +44 171 504 4413, Fax: +44 171 387 1397

Membership Secretary: Dr. Sara Jones, School of Information Sciences, University of Hertfordshire, Hatfield, AL10 9AB. S.Jones@herts.ac.uk, Tel: 01707 284370, Fax: 01707 284303

Industrial Liaison Officer: Dr. Orlena Gotel, Systems and Software Engineering Centre, Defence and Evaluation Research Agency, St. Andrews Road, Malvern, Worcestershire, WR14 3PS. olly@hydra.dra.hmg.gb, Tel: 01684 894674

Publicity Officer: Dr. Andrew Vickers, Department of Computer Science, University of York, Heslington, York Y01 5DD, andyv@minster.york.ac.uk, Tel: 01904 434727, Fax: 01904 432708.

Events Officer: Carol Britton, Department of Computer Science, University of Hertfordshire, College Lane, Hatfield, UK. AL10 9AB, c.britton@herts.ac.uk, Tel: 01707 284354, Fax: 01707 284303

Newsletter Editor: Dr. Steve Easterbrook, NASA/WVU Software IV&V Facility, 100 University Drive, Fairmont, WV 26554, USA. steve@atlantis.ivv.nasa.gov, Tel: +1 (304) 367-8352, Fax: +1 (304) 367-8211

Newsletter Associate Editor (Features and Book Reviews): Dr George Spanoudakis, City University, Department of Computer Science, Northampton Square, London EC1V OHB. gespan@cs.city.ac.uk, Tel: 0171 477 8000 ext. 3701, Fax: 0171 477 8587.

Newsletter Associate Editor (Features and Event Reviews): Ian Alexander, 17A Rothschild Road, Chiswick, London W4 5HS. iany@easynet.co.uk. Tel: 0181-995 3057

RE-Creations

How to contribute to RQ

Please send contributions to Steve Easterbrook (steve@atlantis.ivv.nasa.gov) before the publication deadline. Submissions must be electronic copy, preferably plain ASCII text. A list of the kinds of contributions we welcome can be found in the January 1996 newsletter, or on the web at:

http://research.ivv.nasa.gov/~steve/resg/rq5/ReCreations5.html

	Copy deadline
Issue 16 (January)	18th December 1998
Issue 17 (April)	26th March 1999