

Hypermedia Systems Development Practices: A Survey

Michael Lang, National University of Ireland, Galway

Brian Fitzgerald, University of Limerick

Are hypermedia projects guided and controlled by ad hoc, "quick & dirty" techniques? This survey shows that talk of a "crisis" is largely unfounded.

There's been a flurry of interest lately in the design of Web-based systems, much based on the premise that, by definition, they're fundamentally different from so-called traditional systems.¹⁻³ However, in simple terms, the Web is just another distributed client-server architecture. Many traditional applications such as Lotus Notes databases have become Web-enabled in recent years. As such, they can often simply be plugged in and migrated with little if any redesign other than some code tweaking.

Accordingly, merely calling a system "Web-based" doesn't necessarily imply that its software design differs much from that of a traditional system.

Only when Web-based systems assume hypermedia functionality do they become substantially different from a design perspective. Hypermedia technologies support much richer user interfaces, more complex navigation mechanisms, and more varied forms of information than traditional computer systems (see the related sidebar). In recent years, hypermedia systems (particularly Web-based ones) have grown in complexity and scope as they've begun to involve critical organizational activities such as customer support, sales and marketing, and technical support. So, issues similar to those encountered in traditional development have emerged, such as how to manage requirements, control development processes, coordinate collaborative design, and effectively manage projects.

Due to hypermedia developers' apparent lack of discipline and the technologies' ostensible newness and dynamism, some authors have expressed concern about the delivered systems' quality. Thus another episode of the hackneyed "software crisis" debate has commenced, as critics draw analogies between hypermedia systems development now and the general state of practice back in the 1960s:^{4,5}

Hypermedia development is currently at the stage software development was at thirty years ago. Most hypermedia applications are developed using an ad hoc approach. There is little understanding of development methodologies, measurement, and evaluation techniques, development processes, application quality, and project management ... We are potentially about to suffer a hypermedia crisis.⁶

Although the causal link between systematic development approaches and ultimate project

success is tenuous, critics blame this alleged crisis on the lack of discipline. Charges of malpractice abound in the academic literature:^{5,7}

*In many cases, the development approaches used for Web-based systems have been ad hoc, reminiscent of early days of application software development...Overall, software development for the Web lacks rigor and a systematic approach.*⁸

However, scant objective evidence supports such claims. Thus far, little wide-scale empirical research into hypermedia systems development has been done; to our knowledge, the mainstream literature includes only four surveys of hypermedia development prior to this one.^{2,9-11} So, we decided to find out if the above assertions accurately reflect the reality of practice.

Research objectives and method

We decided to explore

- The extent to which the problems characterizing the alleged hypermedia systems development “crisis” actually exist in practice
- Which, if any, mechanisms developers use to guide and control hypermedia systems development

We conducted a dual-mode survey, by traditional mail and the Web, of purposefully selected hypermedia developers across Ireland. As is typically the case with organizational surveys, defining an accurate sampling frame was difficult. The base population included software developers who specialized in Web or hypermedia systems development, have branched out into “new media” from traditional media, or whose companies possess internal IS departments (for example, financial services firms and banks). We compiled the initial sample from numerous classified industry databases and then systematically reduced it (based on descriptions of activities and work portfolios from Web sites and secondary data sources) to include only those who developed or were likely to have developed hypermedia systems. Three introductory questions ensured that only those respondents with reasonable experience in developing hypermedia systems of substantive scale and complexity were considered in the data analysis.

Our filtered sample consisted of 438 organizations. Sixty-five had ceased to operate or had

“Web-Based” or “Hypermedia”?

Although the Web is a primitive, low-level hypermedia implementation, it's nevertheless the most common hypermedia systems platform today. Notably, the ACM Special Interest Group on Hypertext, Hypermedia, and the Web now goes by the acronym SIGWEB. Likewise, this article considers Web-based systems within the broader umbrella of hypermedia systems, and we prefer the term *hypermedia systems* rather than *Web-based systems*. Hypermedia—a more timeless concept—embraces technologies that predate the Web (such as online help and encyclopedia CD-ROMs) as well as those that succeed the Web (some interactive TV applications, for example). Alternative terms such as *interactive digital multimedia* don't imply the same degree of interactivity or information richness and include applications that we wouldn't regard as constituting hypermedia (for instance, computer games and simple menu-driven movie DVDs).

Our working definition of *hypermedia* is any interactive software system that permits a user to navigate through hyperlinked information by means of various user-selected paths—for example, interactive Web sites, electronic catalogs, intranets, interactive e-commerce systems, online news and information services, interactive courseware and training materials, and complex cross-referenced documentation (such as online help and project dossiers). Hypermedia applications have these standard features: they're database-driven, they integrate with back-end systems, they dynamically generate pages, and they frequently change content. Optional features include rich multimedia content, personalized content, and adaptable user interfaces.

inadequate hypermedia systems design experience. We received 167 usable responses from the remaining 373 organizations after two follow-up rounds, giving a usable response rate of 45 percent. We solicited only one response from each organization; because many were quite small, we felt that asking for multiple responses might cause them to refuse outright.

The study's findings

The cover letter requested that someone in a design role complete the questionnaire, our rationale being to capture a random cross-section of the various design disciplines that contribute to hypermedia systems development. Many hypermedia designers are from backgrounds other than traditional software development, such as graphic design, media production, information science, and technical writing. With the support of content management tools, commercial off-the-shelf products, open source solutions, prefabricated components and applets, and visual programming interfaces, designers from these less technical backgrounds have become capable of developing complex hypermedia applications without needing to know much about

Table 1**Extent of typical “software crisis” problems in hypermedia systems development (as a percent)***

	<i>n</i>	No problems	Minor problems	Moderate problems	Major problems
Controlling project scope and feature creep	161	1	39	43	17
Coping with volatile and changing requirements	164	2	38	47	13
Preparing accurate time and cost estimates	156	4	44	45	7
Coping with accelerated pressures of “Web time” development environment	140	14	56	26	4
Controlling and coordinating project tasks	164	12	64	21	3
Managing communication between team members from different professional backgrounds	166	14	62	22	2

*A Kruskal-Wallis test to compare respondent groups revealed no significant differences.

programming. It’s therefore expected that design approaches and paradigms might vary accordingly, as designers adapt their native disciplines’ traditional processes to the new hypermedia domain.

We asked respondents to indicate their professional discipline (in an open-ended question) and to separately grade their knowledge in a variety of listed disciplines. While examining these responses, we identified three groups of respondents:

- Those primarily from a software development background (33%)
- Those primarily from a graphic design background (26%)
- Those with comparable proficiency levels in software development and graphic design, many of whom designated themselves as “information architects,” “Web developers,” or “Web designers” (41%)

The responding organizations’ primary businesses also reflect this diversity of backgrounds: Web development (26 percent), IT and software development (14 percent), graphic design and media production (10 percent), multimedia development (7 percent), portals (7 percent), interactive communications, branding, and advertising (6 percent), e-learning and computer-based training (5 percent), financial services (5 percent), management consultancy (5 percent), and miscellaneous (14 percent).

The organizations’ size was as follows: 1–20 employees (67 percent), 21–50 employees (10 percent), 51–100 employees (4 percent), 101–500 employees (5 percent),

501–1,000 employees (4 percent), and more than 1,000 employees (10 percent). This distribution profile is typical of industry throughout the European Union, as verified by a Kompass database search (www.kompass.com).

Project management and requirements planning

We asked participants to indicate the actual and planned time and costs of their most recently delivered project of nontrivial complexity. (To avoid speculative responses, we provided a “Don’t Know” category.) We found that 63 percent of projects were delivered in 16 weeks or less, with a median delivery time of 10.5 weeks, consistent with other studies’ findings.^{9,10} At first glance, this supports the notion of “Web time,” an accelerated development environment that’s supposedly characterized by “headlong desperation.”¹² However, our findings don’t suggest any such anxiety; if developers are resorting to desperate measures, they appear to be doing so through intelligent, situated decisions as opposed to rash, unconsidered action. Unconsidered actions are inherently risky and likely to cause problems over time. However, we analyzed the responses to a question which measured the perceived gravity of the problems typifying the alleged “software crisis.” The respondents indicated that they experienced few major problems (see Table 1). This suggests that either these problems aren’t as acute as popularly believed or whatever methods respondents are invoking to tackle them are succeeding.

The two most troubling aspects were controlling project scope and feature creep and

coping with requirements volatility. Not surprisingly, the third most significant problem was preparing accurate time and cost estimates. Though these problems were substantial, projects generally appear to have been effectively managed. Despite the difficulties in preparing estimates, delivery schedules and cost control didn't seem to cause significant trouble (see Table 2). Just 4 percent of respondents regarded "Web time" development pressures as a major problem, and most had few or no problems in controlling project tasks and managing team communication.

As Table 2 indicates, 66 percent of projects were delivered within the agreed budget, and 32 percent were delivered on time, whereas time and cost overruns of more than 50 percent arose in only 17 percent and 3 percent of cases, respectively. It's apparent that more variance occurs in the projects' duration than in the cost. One would expect duration to drive cost, so this is an apparent anomaly. However, no clear distinction was made between "cost" and "price," and the principal reason why project costs vary so little is that most systems appear to have been delivered according to fixed-price contracts. A danger with fixed-price contracts is that actual costs might exceed the quoted price. Given that volatile requirements and scope creep are problematic, project managers committing to fixed-price contracts should factor this in. As a pilot test participant quipped, formulating project plans is easy—it's doing so accurately that's difficult, and things always take longer than you first imagine. For hypermedia development, there's a rapid turnaround period (about three months), development teams are small (four or fewer developers in about two-thirds of cases), and costs generally average about US\$50,000; on this scale, projects have a smaller chance of becoming runaways and spiraling out of control.¹³

We're not arguing that all is well in hypermedia development practice. As Tables 1 and 2 show, a few problems and aspects need improvement. Proponents of the "software crisis" debate could, for example, point out that 99 percent of respondents had problems controlling scope creep, 96 percent had difficulties in preparing cost and time estimates, 68 percent of projects were late, and 34 percent were over budget. However, we don't consider these problems a crisis for two main reasons:

Table 2

Variance in project duration and costs (as a percent)*

	Variance in project duration (n = 137)	Variance in project costs (n = 76)
More than 50% over	17	3
Between 25% and 50% over	22	9
Between 10% and 25% over	24	14
Not more than 10% over	5	8
Exactly on target	29	47
Not more than 10% under	0	8
Between 10% and 25% under	2	6
Between 25% and 50 under	1	5

**(actual - planned) / planned*

- These problems often arise from simple human failings and software development projects' unpredictable nature in general.¹³ It's wrong to present them as aberrations—rather, developers should anticipate and accordingly manage them if they arise.
- The scale of these problems isn't as severe as we would expect a "crisis" to be—where overruns arise, they average about three weeks and US\$6,000.

Capers Jones has remarked that for software development projects in general, "software requirements are usually only about 75 percent defined when design starts."¹⁴ It's therefore quite surprising that 87 percent of respondents indicated that for their most recently completed project, there was a written requirements specification document. However, these specifications weren't as detailed as normal software engineering projects; the lengths reported were fewer than 10 pages (19 percent), 10 to 24 pages (29 percent), 25 to 49 pages (20 percent), and 50 pages or more (32 percent). In response to a separate question, we learned that 63 percent of respondents used documented procedures or guidelines to assist project planning and estimation and 64 percent used them for requirements documentation purposes. These and the aforementioned findings contradict any claims that project management and requirements analysis in hypermedia development is sloppy or opportunistic in general.

Using methods and approaches

Terms such as "method," "process," and "ap-

Respondents widely preferred explicitly documented plans and considered action over ad hoc, just-do-it approaches.

proach” are difficult to define neatly.¹⁵ Inconsistencies in terminology might explain in part why previous research on using methods and approaches in Web and hypermedia systems development is somewhat at odds with our findings. Whereas other researchers reported that none of their respondents used a formal system development method,² this study suggests in contrast that hypermedia systems development is much more disciplined than commonly believed. In reply to a closed multiple-choice question, 84 percent of respondents indicated that their organization used a hypermedia development process that had clear tasks or phases in it. In half of these organizations, these processes were explicitly documented. Only 16 percent of organizations didn’t have a clear process, and the majority of those respondents considered this a problem.

A much more varied picture emerged in response to an open-ended question that asked respondents to list the names of any hypermedia development methods or approaches that they’d used. This question apparently gave rise to some confusion, as we received just 94 responses out of 167 returned questionnaires, 15 of which were too ambiguous to use. Previous research has pointed to the prevalence of in-house methods for Web development,¹⁰ but our survey appears to show that these are mostly not methods in the true sense. Rather, they’re an eclectic, wide-ranging mix of approaches, process models, and toolkits of techniques (usable $n = 79$):

- Hybrid, customized, or proprietary in-house method or approach (23%)
- Traditional “legacy” software development methods and approaches or variants thereof, such as Structured Systems Analysis and Design Methodology (SSADM), Yourdon, Jackson Structured Programming (JSP), System Development Life Cycle, or Waterfall (22%)
- Rapid or agile development methods and approaches, such as Rapid Application Development or Extreme Programming (18%)
- Approaches that focus on the use of tools and development environments, such as PHP, Java, Flash, ASP (Active Server Pages), or J2EE (Java 2 Enterprise Edition) (15%)
- Object-oriented development methods and approaches, such as Rational Unified Process or object-oriented analysis and design (11%)
- Approaches that focus on the use of techniques, such as Storyboards, Flowcharts, Wireframes, or UML (Unified Modeling Language) (8%)
- No method used or development approach is ad hoc (8%)
- Specialized nonproprietary methods for Web and hypermedia systems development, such as Fusebox, Web Site Design Method (WSDM), or Object-Oriented Hypermedia Design Method (OOHDM) (5%)

Because many responses were ambiguous, it was difficult to classify them accurately, and the categories overlap. Quite a few responses indicated that an in-house method was used but didn’t provide any details on its orientation, so use caution in interpreting the table, as the percentages in some categories might be understated.

The top three response categories didn’t surprise us: in-house or hybrid methods, traditional software development methods or variants, and rapid or agile methods. Also, substantial incidences of development approaches occur that are focused around the use of tools, a finding that lends some support to the assertion that developers “delve directly into the implementation phase.”⁷ Notably, little usage of hypermedia-specific methods exists, and it’s significant that the most widely used one (Fusebox) has been devised by a community of practitioners rather than academics.

Despite there being a cohort who admitted to having no process, method, or approach in place, respondents widely preferred explicitly documented plans and considered action over ad hoc, just-do-it approaches (see Table 3). Of respondents, 94 percent agreed that planning is essential, and 80 percent agreed that plans and working methods should be clearly documented. Sixty-nine percent agreed that ad hoc methods generally result in poor systems. The suggestion that “documented working methods are pointless” was firmly rejected by 79 percent. Of course, actual practice might deviate from the developers’ ideals and attitudinal values, a possibility we intend to explore further in future qualitative research.

Regarding using documented procedures and guidelines, 68 percent of responding or-

Table 3**Opinions on aspects of hypermedia design (as a percent)**

	<i>n</i>	Firmly disagree	Disagree	Neutral	Agree	Firmly agree
Ad hoc improvised hypermedia development approaches generally result in systems of poor quality.	153	5	18	8	39	30
To combat system complexity and time pressures, there is an essential need for planning and considered action.	165	0	1	5	32	62
To ensure efficient and effective collaboration in the development team, plans and working methods should be explicitly documented.	165	2	5	13	37	43
Explicitly documented working methods are futile and pointless.	162	40	39	15	3	3

ganizations have them in place for such aspects as technical design documentation, requirements documentation, interface design, system testing, coding practices, and project management.

The study's shortcomings

Our study has a few obvious shortcomings, some of which we intend to address through future work:

- The survey instrument comprised mostly fixed-format questions that captured quantitative data, and responses to the few open-ended questions were scant. A follow-up qualitative field study is under way which aims to elucidate the questionnaire's findings.
- As with most surveys, we had reliability and validity issues. For example, one could argue that the surprisingly positive results were due to the human tendency to make things appear better than they actually are. We took numerous measures to counteract such biasing effects: we thoroughly pilot-tested all questions, assured respondents of anonymity and confidentiality, used multipoint attitude scales to avoid clustering around neutral midpoints, and used split-half analysis to test the data set's internal consistency. Nevertheless, fully eradicating the possibility of bias is difficult.
- It would be interesting to compare developers' appraisals of "successful" projects with customers' corresponding satisfaction levels or to consider if using a development process is associated with higher customer evaluations of, say, fitness for purpose. This survey didn't attempt to answer such questions because of the practical difficulties in designing an appropriate instrument, such as how to select and gain access to "normal" customers.
- We conducted this survey in a small geographical region (Ireland), so we can't generalize our findings globally. The Irish software industry is internationally renowned for its success, and practices might be more advanced and mature in Ireland than elsewhere in the developed world. Cultural norms and attitudes could also be an issue, but this is unlikely, as the Irish software industry is quite cosmopolitan and heavily influenced by practices in the US, UK, and continental Europe. To test for regional bias, the survey could be replicated in another country. We are happy to provide a free copy of the research instrument upon request. It would be especially interesting in the future to conduct a cross-national comparison of hypermedia development practices, but rigorously performing such a study would involve considerable procedural and methodological challenges.¹⁷
- Most organizations in the sample were of small to medium size, as mentioned earlier. This distribution profile is typical of industry across the European Union, but we wonder whether disparities exist between larger and smaller organizations. We ran statistical tests to compare the responses of organizations at opposite ends of the spectrum—those with 1–20 employees (112 organizations) versus those with more than 1,000 employees (17 or-

Techniques are welcome, but they must be adequately validated on real-life industrial-strength projects.

ganizations). The only statistically significant differences were that projects in large organizations take longer (26 weeks on average, twice as long as small organizations) and accordingly cost more. Regarding team size, length of requirements specification, percentage variance in project duration and cost, extent of problems experienced, process documentation, and use of documented procedures and guidelines, we found no other statistically significant differences.


The view that hypermedia development practice is sloppy pervades academic literature and is particularly evident in articles that propose some new—often inadequately tested—method or approach as a universal remedy. Based on our findings, such a view appears unjustified, and cries of a hypermedia crisis seem greatly overstated. In defense, the references we cited earlier that spoke of a crisis date from 1998–2001, which, perhaps by no coincidence, corresponded to the height of the dot-com craze. Many poor performers were forced out of operation in the subsequent downturn. Perhaps Web and hypermedia design has grown up, and practices are more mature now than they were a few short years ago.

Although our results appear to be better than those for traditional software development, we're unaware of any published surveys of such projects that are directly comparable with our study. Indeed, the very notion of conventional software development appears to be going through something of a paradigm shift, with a move toward rapid or agile approaches. We suspect that the practices we reveal are typical of all interactive systems development projects, and that the hypermedia development environment is more similar to than different from conventional software development.

Rather than writing about a largely fictitious hypermedia crisis, the academic community should ask itself whether the real crisis lies in the communication gap between academia and industry.¹⁶ Lately, concern has been growing about IS research's relevance and limited contribution to practice thus far. Already

an overabundance of systems development methods exists in the academic literature, many of which are arcane, impractical, and unworkable. Academics are enthusiastically contributing to this “methodologies jungle” on an ongoing basis, and hypermedia's recent emergence has sparked talk of a “pressing need for new methods and tools.”¹

It's doubtful that a genuine need for new methods exists. Just because a system is based on hypermedia technologies doesn't mean you have to develop it with an altogether new or different approach. As this article shows, you can readily adapt traditional methods and techniques. Though many hypermedia-specific methods are set forth in the academic literature—such as Relationship Management Methodology (RMM), OOHD, WSDM, Enhanced Object-Relationship Model (EORM), and World Wide Web Design Technique (W3DT)—our findings reveal that just 2 percent of respondents had ever used any of these methods and only another 5 percent were otherwise aware of them. We could surmise that the low usage of academic methods could be attributed in part to a lack of awareness or inertia among practitioners. However, the answer probably lies elsewhere. An earlier survey found that understandability, ease of use, and widespread acceptance and reputation among developers are major issues in method selection.¹⁰ In all these regards, most hypermedia-specific methods have serious deficiencies.

The visibility of world-accessible Web-based hypermedia systems has made it evident that many development projects are resounding successes. Once again, practitioners are successfully tackling hypermedia development's new challenges without recourse to the intervention of academic solutions. Any prejudicial notions that practitioners are floundering in a crisis should therefore be treated with great suspicion. Across time, practice has often led the way and informed theory, rather than vice versa. If academic researchers want to make useful contributions to hypermedia development practice, perhaps the best place to start is by learning from practice through grounded empirical research. New and innovative hypermedia development approaches and techniques are welcome, but it's imperative that they be adequately validated on real-life industrial-strength projects. 

References

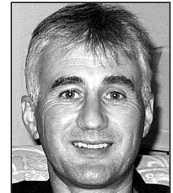
1. S. Murugesan et al., "Web Engineering: A New Discipline for Development of Web-Based Systems," *Web Engineering: Managing Diversity and Complexity of Web Application Development*, S. Murugesan and Y. Deshpande, eds., Springer-Verlag, 2001, pp. 3–13.
2. N.L. Russo and B.R. Graham, "A First Step in Developing a Web Application Design Methodology: Understanding the Environment," *Methodologies for Developing and Managing Emerging Technology Based Information Systems: Information Systems Methodologies 1998, 6th Int'l Conf. Information Systems Methodologies*, A.T. Wood-Harper, N. Jayaratna, and J.R.G. Wood, eds., Springer-Verlag, 1999, pp. 24–33.
3. R.L. Glass, "Who's Right in the Web Development Debate?" *Cutter IT J.*, vol. 14, no. 7, 2001, pp. 6–10.
4. I. Aedo and P. Diaz, "Applying Software Engineering Methods for Hypermedia Systems," *ACM SIGCSE Bull.*, vol. 33, no. 3, 2001, pp. 5–8.
5. O. De Troyer, "Audience-Driven Web Design," *Information Modeling in the New Millennium*, M. Rossi and K. Siau, eds., Idea Group Publishing, 2001, pp. 442–461.
6. D. Lowe and W. Hall, *Hypermedia and the Web: An Engineering Approach*, John Wiley & Sons, 1999.
7. F. Coda et al., "Towards a Software Engineering Approach to Web Site Development," *Proc. 9th Int'l Workshop Software Specification and Design (IWSSD 98)*, IEEE CS Press, 1998, pp. 8–17.
8. S. Murugesan and Y. Deshpande, "ICSE 99 Workshop on Web Engineering," *Proc. 21st Int'l Conf. Software Eng. (ICSE 99)*, IEEE CS Press, 1999, pp. 693–694.
9. P.R. Vora, "Designing for the Web: A Survey," *ACM Interactions*, vol. 5, no. 3, 1998, pp. 13–30.
10. C. Barry and M. Lang, "A Survey of Multimedia and Web Development Techniques and Methodology Usage," *IEEE Multimedia*, vol. 8, no. 2, 2001, pp. 52–60.
11. D.B. Lowe and J. Eklund, "Client Needs and the Design Process in Web Projects," *J. Web Eng.*, vol. 1, no. 1, 2002, pp. 23–36.
12. L.L. Constantine and L.A.D. Lockwood, "Usage-Centered Engineering for Web Applications," *IEEE Software*, vol. 19, no. 2, 2002, pp. 42–50.
13. C. Jones, "Patterns of Large Software Systems: Failure and Success," *Computer*, vol. 28, no. 3, 1995, pp. 86–87.
14. C. Jones, "Determining Software Schedules," *Computer*, vol. 28, no. 2, 1995, pp. 73–75.
15. J.L. Wynkoop and N.L. Russo, "Systems Development Methodologies: Unanswered Questions," *J. Information Technology*, vol. 10, no. 2, 1995, pp. 65–73.
16. R.L. Glass, "Revisiting the Industry/Academe Communication Chasm," *Comm. ACM*, vol. 40, no. 6, 1997, pp. 11–13.
17. M. Lang, "A Strategy for the Use of Web-based International Surveys in Information Systems Research," *Proc. 1st European Conf. Research Methodology for Business and Management Studies (ECRM 2002)*, Management Centre Int'l Ltd., 2002, pp. 187–196.

About the Authors



Michael Lang is a lecturer in information systems at National University of Ireland, Galway, and is pursuing a PhD in information systems at the University of Limerick. His research interests include methods, approaches, and techniques for information systems development. He received his MSc in applied computing from National University of Ireland, Galway. He is a member of the IEEE and the Association for Information Systems. Contact him at the Dept. of Accountancy and Finance, National Univ. of Ireland, Galway, University Rd., Galway, Ireland; michael.lang@nuigalway.ie.

Brian Fitzgerald holds the Frederick A. Krehbiel II Chair in Innovation in Global Business and Technology at the University of Limerick and is a Science Foundation Ireland Investigator. His research interests include information systems development, agile methods, and open source software. He received his PhD in information systems from the University of London. Having worked in industry prior to taking an academic position, he has more than 20 years' experience in the information systems field. Contact him at the Dept. of Computer Science and Information Systems, Univ. of Limerick, Limerick, Ireland; bf@ul.ie.



What have we *done* for you lately?

We publish *IEEE Software* as a service to our readers. With each issue, we strive to present timely articles and departments with information you can use. How are we doing? Send us your feedback, and help us tailor the magazine to you!

Write us at
Software
 @computer.org

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.