Editorial Preface

Learning from the Past; Designing for the Future

Linda V. Knight, DePaul University, USA

The largest design error in the history of computing provokes little more than a yawn these days. It is over; history. The case of the millennium software bug is closed. Yet, as John W. Garner noted, “History never looks like history when you are living through it.” Now that time has blessed us with some perspective, we have a unique opportunity to look back with relatively little bias, revisit the origins of the problem and consider how the bug eluded management controls and what environmental factors facilitated its widespread acceptance in IT departments around the world. In doing so, we can learn something about the present. For example, consider today’s analysts who, with minimal knowledge of the law, regularly make data decisions when implementing “Sarbanes-Oxley compliant” ERP systems. A brief step into the past suggests that these analysts may not be particularly different from the analysts of the last century who made decisions about date formats.

The roots of the millennium bug go all the way back to early computer hardware. During the 1960s, punched cards limited systems designers to a mere eighty characters per record. In this environment, saving two characters by carrying a two-digit year instead of a four-digit year was both logical and common throughout the industry. However, by the 1970s, punched cards were rapidly becoming obsolete, and with them went stringent limits on record size. Yet, in many IT departments, two-digit years persisted. Part of the reason for this was that at first, punched card systems were simply migrated to tape and disk, and the old file design was used, without any thought given to redesign. Even when new tape or disk systems were designed, these new systems often fed data to legacy systems, and the interface between the two systems was simplified when fields were formatted identically. After punched cards finally disappeared, some analysts simply failed to question the two-position year “wisdom” of the past when designing new systems. Others, as late as the mid-1990s, determined that expanding the date field and interfacing with existing two-position year systems would just require too massive an integration effort. Sometimes, decisions to carry the abbreviated two position year fields were justified at the time based upon hardware expense. Using a full four-position year would have increased system requirements for disk storage. Systems storing and processing large numbers of records, and having significant numbers of dates on each of those records, were able to reap the greatest savings by using two position years. Unfortunately, these savings were short-term. Of course, some of the systems built with two position years were themselves short-lived. If a system did not survive to the year 2000, then its analysts clearly made the optimum decision when they chose to use two position years. Unfortunately, this more often than not was a serendipitous occurrence, rather than the result of a thoroughly considered risk assessment.
Although years later, the decision to use a two-position year field led to a significant business problem, at the time this decision was made, it appeared to be a technical design issue, of no interest to management. Thus, a decision having substantial long-range economic and business ramifications was made by relatively low-level technical professionals with little or no management involvement. The designers making this decision were generally quite inexperienced. When the millennium bug was born, the computer industry was young. There was little history upon which to base future judgments. Further, when designers are young, their decisions grow, at least in part, out of the naiveté of youth. To a young man or woman in their twenties, a person who has not experienced the aging of a computer system, the concept that something they create on a computer today might still be around twenty or thirty years hence seems patently absurd. It is no wonder that these young designers joked at the time about having to retire before the year 2000; the year when they knew many of the programs in their world would cease functioning.

In summary, the millennium bug was born when analysts put short-term cost considerations, rooted in historical hardware capacities, ahead of long-range concerns. The youth of the industry and the inexperience of many designers influenced this decision. Analysts designed their systems using traditional date formats. They allowed for capacity growth, but failed to allow for the predictable rollover to a new century. The bug was not caught in traditional management review processes because a long-range business decision was mistakenly treated as though it were a low-level technical issue.

If we are to avoid such mistakes in the future, then we must make simple but important changes in the way we design our systems, databases, and networks. First, we need to require analysts to re-engineer, not for the present system load plus some growth factor, but for the future of the organization. Minimally, this means that all major system, network, and database designs should include, as part of their risk management assessment, an organizational impact statement. This statement should detail the often hidden assumptions that lie beneath the design. The statement should also project potential future ramifications of these assumptions, and detail the worst-case scenarios that might develop sometime after the system is implemented. Preparation of such a statement should not be an afterthought, an annoying piece of required documentation completed at the last possible moment. Rather, we need to make analyzing for future organizational impact an inherent part of the design process.

Requiring designers to actively consider the future ramifications of their work is critical, but if we stop there, we are unlikely to avoid millennium-style design flaws in the future. Instead, we must go on to require that the design's organizational impact statement be reviewed and endorsed by management. This does not mean management audit of design specifications. Rather, it means giving managers the opportunity to evaluate the underlying assumptions built into the design, in light of the organization's longer-range objectives and well being.

Finally, as we form teams implementing new technologies, we need to recognize the value in both the enthusiasm of youth and the wisdom of experience. From youth, we gain openness to new ideas, and a sense that the difficult is truly possible. One need only look at a team of young Web designers to see the joyous embracing of possibilities that youth brings. At the same time, however, we must recognize that youth brings with it a certain naiveté, rooted in a lack of experience. Those who have seen many systems implemented have gained insights into potential pitfalls. They also have developed an understanding of time in a systems sense, enabling them to more effectively evaluate design decisions from a long-range perspective. Thus, when we form teams to implement new technologies, we should try to achieve a blend, a sort of stew pot if you will, where unbridled enthusiasm for new technologies can simmer along side the seasoning of systems past.
Linda V. Knight is associate dean of DePaul University's School of Computer Science, Telecommunications, and Information Systems. She is also director of DePaul CTI's Center for the Strategic Application of Emerging Technologies (SAET), a CTI research group that explores leveraging new and emerging technology within organizations. She teaches and conducts research in the area of Information Technology strategy, development, and implementation. An associate editor of the Information Resources Management Journal, she also is editor-in-chief of the Journal of IT Education, as well as past president and fellow of the Society for the Advancement of Information Systems, an affiliate of MBAA International. She is on the editorial advisory board of the Journal of Cases on Information Technology (JCIT), and is also a member of the Information Resources Management Association Executive Council. She is vice president of MBAA International. An entrepreneur and IT consultant, she has held industry positions in IT management and quality assurance management. In addition to a PhD in computer science from DePaul University, she holds a BA in mathematics and an MBA, both from Dominican University.