Y2K: a retrospective view

Anthony Finkelstein
University College London, Department of Computer Science, Gower St. London WC1E 6BT. (a.finkelstein@cs.ucl.ac.uk)

Introduction

It is now well known that the widely predicted catastrophic consequences of Y2K did not occur. It is also clear that estimates of the nature and severity of the problem were wildly exaggerated. The UK in particular appears to have been subject to an unwarranted public panic. This has, I believe, damaged the credibility of the profession. Though many wilder prophets of doom would not be considered to be engineering professionals, we failed to clearly and effectively counter their claims.

While it would be fun to repeat the more ludicrous of the predictions and hold those responsible up to ridicule this serves very little purpose. It is much more interesting to reflect on how and why the hype and panic spiralled out of control, as a basis for ensuring that such a problem never occurs again.

This commentary is divided into 5 parts: a discussion of the reactions to Y2K post-January 1st; some comments on the changing reactions to Y2K in the period immediately leading up to January 1st; some popular millennium myths; an analysis of why different parties to the Y2K panic acted as they did; and, some suggestions for the future. The commentary is largely polemical. I am concerned that, as soon as possible, the issues discussed receive scientific attention and that the analysis which was so lacking in the Y2K lead-up is undertaken.

Post January 1st

Predicting disaster is, in general, a win-win proposition. If the worst happens then you are a prophet in your own time. If, on the other hand, nothing happens it was only due to your foresight that the terrible consequences were averted. The latter view was certainly dominant in discussions immediately following the uneventful, from a computing perspective, millennium. Of course, the identical argument could be made by the salesman who persuades you to purchase tiger repellant and presents the fact that you are not subsequently eaten by a tiger as proof of the efficacy of the repellant.

A suggestion which received wide currency after January 1st was that disaster would strike — but not on that date. A variety of different candidates were suggested. The first day back at work, when the Y2K contingency team were stood down, the end of January, February 29th, the end of the financial year, and so on. Putting back the end-of-the-world is, of course, the standard fall back of millennialists of all types. These revised prophecies have proved as mistaken as the one they replaced.

Two other reactions have been evident. The first says that widespread failures did, in fact, occur but that the commercial interests and the need for secrecy to protect client confidence, for example in banks, meant that there was no publicity for these failures. This is a standard conspiracy theory explanation and can be readily dismissed. More seriously others point to the small number of widely publicised failures as evidence that their predictions were correct, "after all, what might have happened if ..." The examples given are a failure at Oak Ridge Laboratories in the USA (which processes nuclear weapons), failures in the alarm systems at Japanese nuclear power stations, the failure of dialysis machines in a Scottish hospital. These are serious sounding but what characterises all of them is the absence of a detailed account of what actually happened; ignorance about the general failure characteristics of the systems (how often do false alarms occur generally); and, the fact that they were dealt with quickly and effectively without adverse consequences. The examples are in any case few and far between despite the fact that the world media were eagerly seeking such failures.

Pre January 1st

In the period leading up to January 1st those who had made dire predictions of catastrophe proved amazingly unwilling to adjust their views in the face of what was actually happening. A good example of this was September 9th 1999 (9/9/99). On this date data marked "never to expire" (realised as expiry 9999) would be deleted bringing major problems. This was supposed to be a pre-shock that would prepare the way for the disaster of January 1st. Nothing happened. Now, if you regarded the problem as a serious threat in the first place, this should surely have acted as a spur to some serious rethinking. It did not.

September 9th 1999 is only an example. Similar signs should have been evident on January 1st 1999, the beginning of the financial year 99-00, December 1st, and so on. Indeed assuming, as was frequently stated, poor progress had been made on Y2K compliance programmes we would have anticipated that such early problems would be common and severe. I see no reason to suppose that problems should not have been more frequent (or at any rate as frequent) in the period leading up to December 31st 1999 than afterwards given that transactions started in 1999 may complete in 2000, while after January 1st new transactions start and finish in the new millennium.

The ability to distinguish signal from background is an elementary engineering skill. Most real software systems are full of errors, they fail frequently. Any failure or problem has to be measured up against the background rate of failures that users experience. This same issue of background applies to Y2K expenditure. Large figures - billions (!) of dollars - were given as the cost of Y2K compliance in certain industries. These figures while absolutely large are relatively small when set against IT expenditure overall and even smaller when set against the financial and other economic transactions that computing systems mediate.

Millennium Myths

It is perhaps worth spending a little time reviewing some popular millennium myths.

"At the stroke of Midnight". However appealing the idea of a millennium meltdown might have been to the media it owed more to popular culture than technical analysis. Ludicrous war rooms and standby arrangements cost business large sums, wholly unjustified by any realistic appraisal of the potential failure modes of the systems they operated.

"Embedded systems are out to get you". Much of the worst Y2K hype surrounded embedded systems. Only a tiny proportion of such systems have a clock at all and a very small subset of these use absolute as distinct from relative time. This is common knowledge. Despite the dreadful predictions, no serious failures occurred.

"If <fill in blank> are spending <ludicrous amount> then it must be a serious problem". Firstly, the poor decision making of somebody else is not good grounds for me to copy it. Particularly as large organisations which make sane and rational decisions about their IT expenditure are rather thin on the ground. Despite this, and I wish to emphasise this point, I believe that most organisations that spent heavily on Y2K were justified in doing so! Not because of Y2K itself but rather because an organisation that does not know which are its business critical systems, that has no contingency plans, that does not know what systems it has and has failed to document its software is headed for trouble - regardless of date. Any efforts to address these problems are worthwhile. Y2K was, for the most part, a red herring.

"But they are only <some modest percentage> of the way". Throughout 1998 and 1999 we heard persistent complaints about the delays in addressing Y2K compliance issues and associated predictions that this was likely to fuel disaster. Key industries were said to be far behind with their remediation programmes. Apart from the fact that, in the final analysis, most of these industries successfully completed their programmes this complaint ignored the well known 80:20 rule. Errors are not randomly distributed. Only a relatively small amount of effort, well directed, is necessary to avert a very significant proportion of the risk.

"It’s not us it’s them". In predictions of Y2K mayhem, ripple failure of distributed systems formed an important component. Connected systems would fail with knock on consequences - like a line of dominos. This suggestion ignored the fact that distributed systems are built precisely because they are more stable and can cope with failure. By providing components, integrity checking at interfaces and independent computational resources, such systems are inherently much less failure prone than conventional systems. Ripple failure is a very rare failure mode.

"What will happen if". Most real computer users are used to failure, after all they are quite familiar with it. If their computer does not operate they use well practiced standby arrangements. If their computer makes a ludicrous suggestion they ignore it. If a funny date is printed on an invoice they tipex it out. In any case just because something could happen it does not make it likely to happen a distinction which most computer users understand but appears to have been lost on most so-called Y2K experts.

"It's not us that I worry about, it is <fill in name of less developed country>". Of course, it is now well known that, despite devoting much less effort to Y2K than the UK, most other countries (including less developed countries - such as Italy) experienced no significant Y2K problems. The explanation for this has been variously ascribed to the fact that either they have less technology or that they have more newer technology. Perhaps the most obvious explanation which is that they got it right and we did not, is too difficult to accept.

Who and Why

In order to understand Y2K reactions properly it is worthwhile to consider the major groups concerned.

Foremost of these is the public. Despite the best efforts of IT educators the public is in general ignorant about computing. This ignorance is combined with a growing recognition that they are dependent upon it. The mixture of ignorance and dependance is fertile ground for fear, it is perhaps no wonder that the level of public panic exceeded what would have been reasonable.

Many business managers have never considered themselves to be in technology dependent businesses. The growth in IT use in their businesses happened, relatively, slowly. Y2K and the attendant publicity caused a sudden jolt. The realisation rapidly dawned that their businesses were almost completely dependent upon a technology which they neither understood nor felt they had adequate control of. This shock again caused a panic reaction.

IT managers played an important role in Y2K hype. Many IT managers used Y2K as a loophole to fund long term maintenance, legacy and contingency planning problems which they would have had to address in any case. Similarly the advantages of being able to purchase new equipment and software with budgets drawn from outside the normal capital approval processes are obvious. As argued above this may have been mildly dishonest but was not harmful. On the other hand the funding required them to make a Faustian pact in which they promised that Y2K was a one-shot problem rather than educating the business in the long term need to invest in appropriate management of their software assets. It is well known that many centralised IT Departments have never felt entirely happy with distributed computing and with user control over the PCs on their desk. For some of these IT Departments Y2K has provided an opportunity to temporarily stem the tide and reassert their control.

The role that consultants and so-called Y2K experts have played is a complex one. Clearly money has been a substantial incentive to turn up the burner under Y2K hype. More significantly there has been a serious loss of a sense of proportion caused by being too close to the problem. I strongly suspect that many consultants recognised that potential consequences of Y2K had been exaggerated but felt that, on balance, the panic was good for business. In other words that unless business managers were made to panic they would do nothing about the broader issues of long term maintenance, legacy and contingency planning. This attitude is deeply wrong, unethical and sidesteps the overriding obligation to educate management to understand that software costs much more than its purchase price.

It is a source of continuing surprise to me that academics played so little part in Y2K discussions. There is virtually no serious scientific literature on the problem. For the most part, while the problem occupied the IT industry, academics either adopted a stance of ironic detachment or viewed it as too convenient an illustration of the importance of effective software engineering to be overly concerned with the facts of the matter.

It might have been expected that science policy experts would comment sensibly on Y2K. Like the Governments they advise however, they have been too badly burnt by the BSE issue. In short the political costs of under reaction were greater than the perceived political costs of over reaction. So, they over reacted. Such an over reaction was made easier by irresponsible press coverage. Y2K was too good a story to be spoilt by a reasoned risk assessment. It had the right "shape", it combines technology, hidden risk, lurid consequences. It is even an opportunity to poke fun at clever-dick computer experts. It was, as a journalist friend told me, "too good to check".

Irrationality is much more common than engineers and scientists like to think and feeds deep cultural undercurrents. Society was primed for millennial angst and Y2K fitted the bill perfectly. Religious nutcases and political extremists fed this angst. The Internet provided an excellent medium in which it is often impossible to distinguish the serious analyst from the irresponsible propagandist.

The Future

We need to learn from Y2K. As a profession we need to take greater responsibility for educating the public and business management about the issues we face - without resort to crude simplifications. We need to ensure that politicians and the media hear a balanced professional voice - basing its opinions on an empirically founded attitude to risk. We need to recognise and combat hype - even where it may seem to our short-term professional advantage to allow it to continue. We need to rebuild the confidence of the public and of business. We need to develop the science and practice of software engineering so that we can deal with the underlying chronic problems that Y2K signalled. We need to face up to the fact that we got it wrong.

This is the original (slightly longer) version of the paper that appeared in Computing & Control Engineering Journal, August 2000, v11, N4, 156-159 (with a companion piece by Martyn Thomas).