ANUSHKA DHAR

A203322969

FINDINGS ON Y2K TESTING EFFORT

INTRODUCTION

The main objective of this research is to figure out the testing effort on fixing the Y2K defects. It also discusses about the reasons which let to these defects and what kind of software quality management system was being followed at that time as the testing procedures being followed at that time was not able to find Y2K defect for many years. Moreover, what were the consequent testing measures to solve the defect and how much money was involved in it? Also, I have provided with a SQM environment which could have helped in reducing the chaos created at that time.

Problem Description:

The phrases “the Year 2000 Computer Technology Problem” or “the Millennium Bug” refers to the same problem – a defect that exists in millions of computer programs worldwide that causes erroneous handling of date information if not correct.

The root cause of the Y2K bug could be traced out in the early days of computers. During that time period data storage was quite limited and very expensive so any method which could save storage was readily adopted. Since in early 1950s or 1960s there was no guarantee that how long one software could last, so eventually the dates were stored in two-digit format that is 1955 would be stored as 55. So, some programs could not distinguish between the year 2000 and the year 1900.This eventually lead to some date-related processing to operate incorrectly for dates and times on and after 1 January 2000.Another problem was that some programmers had misunderstood the Gregorian rule which determines whether the years that are exactly divisible by 100 are not leap years, and assumed the year 2000 would not be a leap year. These were the main reasons behind the Y2K problem.

It was assumed that once the date was turned from December 31, 1999 to January 1, 2000, the computers would be so confused that they would shut down completely. Since, by 1999 computers had become an integral part of our lives so the new year was expected to bring serious computer repercussions.

SECTORS EFFECTED:

Since this defect happened due to the format in which year is stored in the system. It effected the following sectors:

- Education sector
- Transportation
- Federal operations
- Law Enforcement
- Local Government
- Medical
- Power and Utilities
- Software

- Business sector as they were losing large amount of money
- Personally owned computers.

REASONS WHICH LED TO Y2K Problem:

Pervasiveness: In every application dates are present in some form including information systems, command and control systems, and weapons systems. They form the basis of derived calculations for such data as age and amortization schedules. Dates are also the basis for a number of sorting routines, error checks, validations, and logic for terminating applications.

Large programs – poor coding practices:
The biggest obstacle to fix the Y2K bug was that a large amount of code had to be checked. One problem was that the original code was developed after lot of months and years and that too by different programmers. So, sometimes few programmers instead of using library routines for dates used to create their own date handling function. Eventually, the Y2K repair had to search frequently a significant portion of the codebase to find all instances of date functions. Since the code was developed years back so the programmers who developed the original code were no longer available. As a result, the company had to hire new programmers which had no idea about the code. Eventually, this lead to increase in time and cost required to fix the problem significantly, since the new programmers had to first understand the code before they could fix it.

Undocumented Code –
Another problem was that there was lack of documentation on the implemented programs. Without the proper documentation the new programmers would have taken much longer time to understand the code before they could even fix it. For a company that was trying to fix the Y2K bug as quickly possible it meant higher costs and tighter schedule.

External Libraries:
For some companies if the programmer used an external library for the date manipulation and the developer who used it is not available then the company has either to but a different library and has to replace the old one or rewrite the library and plug it into the program. Both options are extremely expensive and require lot of time.

Planning Strategies Used:
The key critical success factors for addressing the problem included
- To raise the problem to sufficiently high management level so it gets attention.
- To estimate the cost of solving the Y2K for a particular organization
- Deciding which systems were critical to business, and hence deserve priority, and which systems can be left unaltered for time being.

AWARNESS AND ACTION PLANNING

One of the critical factor to deal with Y2K problem was to raise the awareness about the problem. Although there was knowledge of potential existence of the Y2K problem but a number of organizations were slow in addressing the issue due to following reasons:

- the misconception that the problem will occur on somebody else's watch
- the belief that it will not be severe within a particular organization
- an inability to scope the local impact and develop a plan to resolve the problem

Mostly, high-level managers were underestimating the magnitude of the problem within the organization and though that low-intensity effort would be required to resolve the problem.
So, it was important to make the high management aware about the impact of the problem within the organization and make them fully involved for careful planning estimation and allocation of resources.

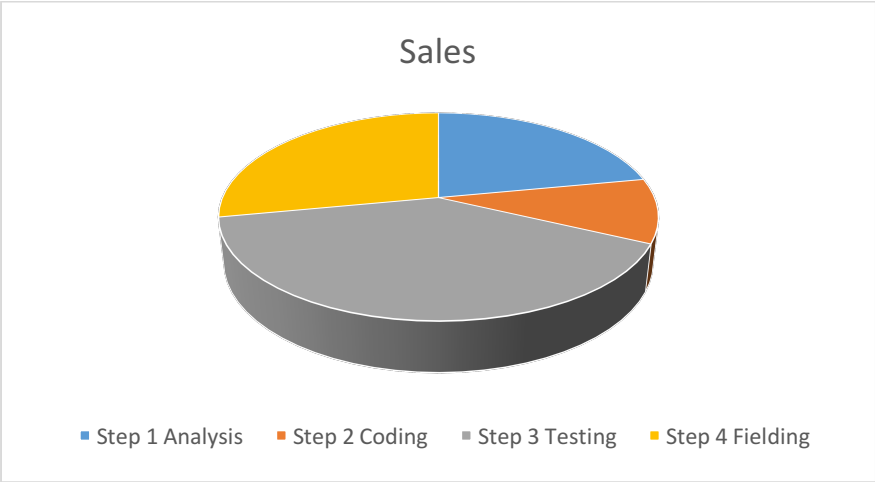
COST ESTIMATION

Y2K bug was very much unpredicted and uncertain in its own way. In very low level, the total money lost can be calculated from modifying the code documents. There are various code snippets which uses the date and time stamps for multiple purposes in various programs. From finding the places to start replacing the error till implementing the new code for the clients needs time and resources. Cost factor associated with this is all affected by the short time the organizations had to fix this issue. They had to rely on resources, man as well as machine power to get ride of Y2K bug quickly. In the other hand, at the higher level, there is cost associated with the systems using the Y2K affected programs. For example, businesses, banks, government institutions, logistics, planning etc, have to compensate their customers for the losses happened.

Mentioned below is some results after the analysis of total loss incurred in various systems due to Y2K bug.

1. Ground and Airborne Radar System ($2.02 - $8.52) per line of code
2. Communication Processing Systems ($1.23 - $ 5.54) per line of code
3. C2 Planning Systems ($1.22 - $1.84) per line of code
4. Logistics Support Systems ($1.02 - $1.39) per line of code
5. Department of Defense, total cost estimate – ($2.4 billion)

The graph below shows the share of total cost incurred during various process of bug fixing.



- It can be noted that almost 35% - 45% of the total cost is associated with the testing phase of the new code. It is because of the following reasons:
- Code needs to be fixed in multiple places
- Other programs with dependencies needed to be debugged or re-compiled
- The total impact area of the bug was not known, so range of testing needed was also uncertain, making the cost more and more.

Also, different cost estimates were also made about the worldwide cost involved with fixing Y2K problem:

- Gartner Group estimated worldwide cost to range from $300- 6 CMU/SEI-97-TR-002 $600 billion, at a rate of $1.10 per executable line of code (LOC).
- Another number from a reputable source is that of Capers Jones, Software Productivity Research Inc. of Burlington, MA. Jones worldwide estimate is $1.6 trillion.
- The difference between the cost estimation in both the cases is that Jones estimation includes over $300 billion for litigation and damages but Garner's doesn't include that.
- According to initial study done by MITRE on the cost estimation for solving the problem within the US Department of Defense was higher per LOC as they had greater percentage of embedded systems

In case of financial services, we can compare the cost spent by Citicorp, General Motors, Bank America, Credit Suisse Group, Chase Manhattan and J.P. Morgan. to solve Y2K issue. These six institutions have estimated to collectively spent over $2.4 billion to solve this bug.

Company	Cost Involved(millions)
Aetna	$139
ATT	$300

Bankers Trust	$180-$230
Cendant	$25
Chase Manhatt.	$300
General Motors	$400-$500
Xerox	$116
Sears	$63
Merrill Lynch	$375

According to Commerce Department estimation to exterminate the Y2K bug form the nation's private and government computers cost around $100 billion.

For federal government it took around $8.4 billion to solve the bug.

Bell Atlantic spent $416 million on Y2K, using it to upgrade hundreds of personal computers and evaluate microchips in thermostats for 3,100 buildings, a necessity because many buildings house temperature-sensitive telephone switching equipment.
If the temperatures are thrown off even on one floor of some of our buildings, it could knock out phone service for thousands of customers," said James Smith, a Bell Atlantic spokesman in New York.

T. Rowe Price spent $44 million, testing and upgrading 3,000 personal computers and setting up a $20 million mock operations center with leased equipment that turned the clocks ahead to Jan. 1, 2000, and ran months of simulations of its investment operations.

Baltimore Gas and Electric Co. spent $46 million to set up a test center, upgrade 5,000 personal computers, evaluate security systems at the Calvert Cliffs Nuclear Power Plant, replace 6,000 meters that track power consumption, and purchase phones and other equipment for a New Year's Eve command center.

Government and hospital budgets have also been hit hard.

Baltimore City has spent $15 million, Baltimore County $9 million, Anne Arundel County $11.7 million, Howard County $3.1 million, Harford County $1.2 million, Carroll County $672,000 and Frederick County a relatively small $375,000.

We can see variations in the cost-estimates according to different organizations and cities that is because of the different strategies the organization has chosen to solve the Y2K problem.

Triage Planning:

For many organizations, it was not possible to solve the problem for all their systems before the system fails, either on 01/01/2000 or before that date depending on the application. For this a triage approach was used in which priorities were set to select the systems which need to be converted and saved, and which to allow to fail.

So for this new business goals and objectives were defined as business value of each system had to be estimated, the risks and liabilities involved if the system would fail and plans and requirements for the future enhancements and potential replacement of the existing system.

Especially, the mission critical system that were not slated for immediate replacement need to be converted regardless of the cost. On the other hand, the system which were of low strategic importance were allowed to be failed if resources were not available.
These decisions were mainly based on the business value, difficulty of conversion, technical risks, resources available and time remaining.
Plans needed to be developed for the systems that were not being converted as the functionality of the system, its business uses, interfaces, and financial, technical and legal impact had to be assessed.
Plans also need to be developed for the redevelopment of the system as the resources became available.

There was requirement of remediation plan to allocate resources to the task and also to develop a schedule for major technical tasks required. Also, the schedule had to be monitored closely because slippage can have serious impact on other Y2K conversion projects.

TECHNICAL STEPS USED TO DEVELOP THE SYSTEM

Following were the steps which were adopted to minimize the risk and maximize efficiency as they are arranged in a systematic flow:

1. Developing a high level system inventory.
2. Developing an Impact Analysis
3. Plan the remediation
4. Perform the remediation
5. Test the changes
6. Migrate to production

Developing A High System Inventory

The most initial step was the evaluation of all the existing system. For this a comprehensive inventory of all software and hardware products was done. In this step a complete inventory was required for PC hardware and software, network software and databases.
Inventory analysis is one of the most critical and important step for solving Y2K problem. It involves identification of the following:

- applications and data sets of the organization
- the object code and libraries used to build these applications
- the source code used to build the object code and libraries

- databases and data files used to generate the data sets
- scripts and command files for building applications and data sets
- parameter, set up and initialization files (e.g., for sorting utilities)
- corporate naming convention policies and plan templates
- other auxiliary and ancillary files

This analysis can take up to two or three months. But while doing these organizations had to face lot of problems as it was hard to identify exact source code that belongs to a particular application. Identifying the source for a library which was not produced in-house also caused problems. Tracking down the older versions was hard and migrating to the attest version of software product can require changes in the operating environment along with the specific Y2K changes. Also when the problem occurred the code was mostly missing for the 20% of the applications.

Developing an Impact Analysis:

Impact analysis helps in determining the location where the date exists in the system, also the modules and statements that are effected and the major critical paths. In some application there may be 30 to 50 different date formats. Since mostly 10-20% of the code contained Y2K defects so it had to be changed. Once the inventory was complete, a group of individual vendors were required to find out what will be the fixes, what will be the costs involved for those fixes, what will be the timeframe for the implementation, if correcting different systems within the same time frame could be coordinated and what changes will the correction would bring to the organization. Further, IS managers had to decide whether to use existing staff to work on the problem or to use outsourcing. After this the manager decides a strategy to face the Y2K problem. For this there are usually two strategies designed one is called the rework strategy and other one is the replace strategy. The rework strategy involves modifying the existing applications, installing patches and rewriting the applications to use the appropriate century in calculations based on data characteristics.

In replacement strategy, the organization has the option to either purchase or to build new system to fit the business needs of the organization or to keep upgrading to the versions that are compliant. This strategy was quite expensive but provided two benefits one is the implementation was much faster and the organization was benefited with the new enhanced system.

Pattern matching tools were used to provide general rules for detecting date fields, date literals and source code that manipulates or calculates dates. Since each organization has their own naming conventions so these rules had to be adjusted for each organization.

Plan Remediation:

Once the magnitude of the problem was analyzed the next step was to develop a strategy for remediation. The basic options used at that time were:

- expansion of the database to accommodate a century indicator

- development of date “windows” that are processed differently depending on whether the date is early or late in a century

- compression of the date fields in the database

Although out of all these three solutions expansion of the database was the most complete solution but it is the most expensive one also.

Database Expansion – It involved changing the database records from a 2-digit year "YY" to a 4-digit year "YYYY". A secondary option was to keep the current date fields and add a century indicator as separate field in he database.

Windowing: In this approach a 100-year window is specified and how dates in specific intervals within that window will be processed. There are following different windowing technique:

A "sliding" window allows the 100-year interval to advance according to a selected criterion.

A "fixed" window uses a static 100-year period that does not change throughout the lifetime of the application.

Using this technique application can be converted at lower cost than the database expansion. They also offer the potential of developing date modules that could be called from within applications to process dates. But the windowing techniques are not permanent and they will not be able to handle the data that span more than 100 years.

Field Compression: The compression approach packs 4-digit data into a 2-digit field. Through it the current database and current fields in the database could be used. But still to process the new format program changes are still required and these changes will need to be applied consistently.

In addition, conversion programs will need to be developed to apply the changed formatting. Once the changes are made, there may be data conversions at runtime, with potential operational performance issues.

Implement Changes:

The main aim of most of the organizations was that all the changes to the application should be done before the middle of 1999 so they could get sufficient time for testing the corrected applications. PC's and the embedded system were the hardest components to identify and correct the Y2K problem.

The organizations considered three options:

- To poll the real time clock at certain interval check the date in RTC and correct it if needed. The system is updated by the RTC and correct it if needed. The system is updated by RTC and most system get their date from the system clock.

- Second approach was to flash upgrade the BIOS on the machine that are found not be capable of handling the Y2K problem.

-Third option was to completely replace the PC's with the newer system. This is the most expensive option for the PC's

Different organizations used different approaches to deal with the problem. Although automated tool can be used to perform the task but at that time some languages didn't have the automated tool support and thus the entire set of transformations need to be done manually.

Testing:

After the required changes were made then the next step was to unit-test and system-test the corrected applications in order to ensure that they were compliant to Y2K issue. Along with that interoperability of the application and ensuring their functioning was required. Another necessary kind of testing which was important was to create a special time based testing environment with time-sensitive scenarios to prevent the possibility of loss of data or system resources. One key point to be kept in mind was that the year 2000 was a leap year so the testing should be inclusive of testing all the leap years to ensure that date calculation was accurate.

Keogh mentions that about "40% of the Y2K- process is testing".

To understand the different kinds of testing used for Y2K Project we will consider the case of Belgian Bank. They created tests both for the dates before the 2000 year and after it. After the Y2K modifications the system was run with the dates of today and results were compared with the results produced by the old system. After the millennium version was put into production. The first phase of testing i.e. "Unit Testing" was conducted in India and included the debugging of the individual programs. The support team in Brussels was responsible for the "System Testing" which determines whether the entire system is operates as intended.

Then the final testing i.e. "Acceptance testing" by users and managers was done to decide whether they accept the system. The test program consisted of "business as usual situations" and different functions within the system are individually tested. The reason behind this thorough "regression" technique was that Y2K changes might also change something in the system that wasn't expected.

A first test above 2000 was conducted by the newly installed test department for dates plus 28 years. The reason behind this was that every 28 years you have the same calendar and results can be compared with these of a system run with dates before 2000.

Additional specific tests were done for the rollover 1999-2000 and for specific dates such as February 29,2000.

It is estimated that Belgian Bank took more than one year and that as many as 50 people of the new test department were involved along with end users and operational people involved in the acceptance testing.

But it is very important to form good testing procedures. And at that time most of the IT organizations didn't have a proper test organization which eventually result in poor testing and many errors and the problems when the modified and new version was implemented and processed. Even the Belgian Bank faced this shortcoming at that time and as a result establish separate testing department for Y2K project and this extended their time to solve the Y2K problem.

Mostly the testing strategies used by different organizations were unit testing, integration testing, system testing and acceptance testing.

IBM also provided a framework for Y2K Testing:

Unit testing and integration testing that are forms of structural testing whose primary goal is to locate errors created during remediation. These testing phases exercise all structures of the system. They include operations testing of normal production scenarios, stress testing of abnormal circumstances and volumes, and recovery testing after system failure.

System testing and acceptance testing that are forms of functional testing whose primary goal is to locate design errors and improper implementation of requirements. The functional testing includes requirements testing, regression testing, error handling, manual support testing, and interface testing.

5)Integrate Corrected Applications:

Once all the above steps were done the each new or corrected application was integrated into the production environment by making the conversions that were needed and planning the type of cut-over method to use. Most organizations kept two good backups before 2000, so in case if nothing works right there would be a good copy of data available. Also along with the functionality improvements for applications, the end-user training and as also planned for d implemented.

Following tasks were performed while integration :

• development of procedures for conversion of data, either automatically or manually

• development of new programs for screens or reports that require the new date formats

• acquisition and installation of required storage devices and other hardware and software that will be needed for the conversion

• development of new job control programs and parameter files

• updating of all documentation s

• backing up of old data files and load modules • conducting parallel testing of new and old system components

• transferring of new components into a testing environment and placing into production on an incremental basis

• monitoring of system performance with each new increment

Time taken to fix the code.

Awareness that Y2K issue as a bug was identified by many organizations from 1996's. Either they couldn't fix completely or started looking into this at the same time as well as not promoting the possible effects of the, ended up in spending too much money in solving the issue. Such companies later had to spend more money due for fixing the issue. The primary cost lost in the process was due to the time delay during the whole process. Apart from this, the organizations during 1998- 99 years, figured out the issue and took quick action plans to study and fix the bug. Irrespective of the start date, from both the cases above, all these organizations could fix and run the new code much before the year 2000 began. The total time to fix it varied for various entities from 2-6 years depending on their size of their company and factors affected by the issue. It could be concluded that the more the impact of the Y2K for a particular company more money needs to be spend for fixing and the same who could invest more money fixed the issue ver fast.

WERE THE STRATERGIES USED EFFECTIVE AND EFFICIENT

From my perspective the strategies chosen by organizations were effective up to a point. The initial steps taken for solving the problem like making high level management well aware about the situation, estimating the cost and triage planning helped a lot in solving the problem to an extent. But some of the steps taken by them were not that effective like high level inventory system was mostly developed by manual work instead it could have been done easily with the help of CM system which helps in identifying sources files easily. Since the measures taken by them involved lot of manual work instead of the automated tools it eventually led to the increase in time to resolve the problem along with lot of expenditure.

I also find that even though the measures implemented by the officials were effective there was a major amount and time spent for doing the analysis and preparing the study plan. This could have been avoided in this case as, the issue required high priority and quick action plan.

STEPS TO IMPROVE:

Since inventory analysis was one of the most step for solving Y2K problem having a configuration management would have eased inventory analysis considerable. Since, identifying the source files through a CM system is usually straightforward. Another way to improve the analysis could be by using automatic and intelligent tools. One of the tool which could have been of great help would be "crawlers" for a specific operating system. A crawler would analyze the entire operating system identify all the executables, construct an inverse building map and locate source files. Most of the inventory analysis could have been automated with the help of this tool. Otherwise, doing inventory analysis manually is an error-prone process.

Impact Analysis:

First to successfully perform remediation we need to understand the code properly and for that parsing technology could have been used.

We need a technology for addressing following needs:

a standard intermediate language, such as that represented by Software Refinery, to serve as an interface between parser and impact-analysis tools, that is sufficiently detailed to handle Y2K analyses and conversions

• a grammar and a parser for every variant of a programming language

• parser generators for every grammar

Once the code is parsed, the date impacts could be analyzed through the following tools :

• a catalog of all known date formats and manipulations occurring in practice and their actual patterns in various languages

• pattern matching algorithms for locating date processing

• forward and backward impact analysis based on slicing technology

• assessment and evaluation of tools, algorithms, grammars, and catalogs

Remediation

To provide remediation following technologies would have reduced the risk :

a catalog of algorithms for widening date formats in various languages

• a catalog of algorithms for windowing schemes in various languages

• a catalog of algorithms for date field compression in various languages

Test Remediation

The area of impact analysis, inventory analysis and remediation only have 30 % of overall life cycle of the Y2K migration project. One of the major goal should be automating the testing phase which comprises of the 50% towards the cost f a Y2K project. As the project progress to the testing phase different test scenarios and coverage guidelines needed . Also, testing is especially important for mission-critical systems.

Requirements of Coordinated Effort ;

Coordination is required both at an organizational level and at a national level.

This agenda needs to include

- development of awareness
- clearinghouses of tools, tool evaluations, and tool vendors
- research program for development of needed tools
- clearinghouse of software certification for Y2K compliance
- sharing of lessons learned and case studies

General process templates

The research on Y2K opens up various questions about processes and functioning of the software run industries. It can be noted that there is not pre determined procedure or solution for various issues, but there can always be an efficient way to deal with the issues by proper planning.

- Ideas/Issues founded should be first prioritized as low, medium or high. This gives as idea about the type of issue.
- Then the impact of the issue should be prioritized. There will be minor issues which have a large impact on code (eg. Y2K itself, had to change the YY to YYYY, but had a wide impact).
- Possible fix and probable implications should be communicated with the higher management. This include the availability of resources, time for whole procedure.
- The management can take the decision for the optimal solution in terms of cost as well as time.
- The solution is passed back to the team, and it can be done by programming and testing carried out alternatively with in the time frame.

The roles can be categorized for the above operations in the following way:

- The analysts – does the study and possible solution for new implementations and issues.
- Managers - deals with the cost estimation, delivery estimation, resource allocation for the whole procedure.
- Programmers/testers – deals with the process implementation. The development/testing phases are done by them.

RESOURCES :

1. http://www.gpo.gov/fdsys/pkg/GPO-CPRT-106sprt10/pdf/GPO-CPRT-106sprt10-6-1.pdf
2. http://courses.cs.vt.edu/professionalism/Y2K/Y2K.Overview.html
3. http://articles.baltimoresun.com/1999-12-23/news/9912230307_1_y2k-computers-new-year-s-eve
4. http://www.doc.ic.ac.uk/~nd/surprise_97/journal/vol4/ak16/
5. http://articles.chicagotribune.com/1998-11-20/news/9811200209_1_y2k-problem-y2k-czar-millennium-bug

6. http://www.freepatentsonline.com/article/Business-Economics/54035911.html
7. http://www.sei.cmu.edu/reports/97tr002.pdf
8. http://scholarworks.lib.csusb.edu/cgi/viewcontent.cgi?article=1203&context=jiim
9. http://www.drdobbs.com/tools/strategies-for-solving-the-y2k-problem/184410551
10. http://www.academia.edu/760099/Calculating_the_cost_of_year-2000_compliance
11. https://en.wikipedia.org/wiki/Year_2000_problem
12. http://www.cs.swarthmore.edu/~eroberts/cs91/projects/y2k/Y2K_Errors.html