

LEAD TIME REDUCTION FOR PROJECT MANAGEMENT

Dr. Bhargav Gangadhara,

¹Senior Technical lead/Director, Jack Henry and Associates, USA

¹ bhargavmtechmem@gmail.com,

Abstract— The present day market deals with speed in delivery of products, customer satisfaction. One thing which flashes our mind is lead time or cycle time reduction and resource planning for in time manufacturing and delivering of products to the customers.

Organization is into business of providing the solution to mobile phones. This consists of development team, Internal testing team, validation team and factory quality check. The testing group has majorly two tasks, one is testing and the other is to release to downstream user.

While testing one of the problem faced by the group is manually executing all test cases which is reduced with the aid of an automation tool which replaces the manual testing of mobile phones. Understanding the mobile model and alternatively writing a code for sanity testing, key press, and key map will help the testing team to test all the test cases and finally reduce the number of rejection per release. As the number of defects reduce in first loop itself i.e. at internal testing itself the time, cost and quality involved for further testing is reduced this will in turn reduce the cycle time.

The next task is resource planning of resources to release to downstream user, which takes a lot of bandwidth of testing group to upload the document and source code. To predict and optimize overall cycle time of server based on understanding of upload sequencing and server load and analysis of bottle necks in network. To meet the current load which will save several million dollars as the server cost is high.

KEY WORDS: Cycle time reduction, Manual testing, Automation, Resource planning, Server upload.

1. INTRODUCTION

As a software quality assurance percipient the major concern was to identify a critical area for research on internal quality management. Even though concept seems to be simple on papers practically performing takes some pain, let us understand the organization structure. The organization is into business of providing the solution to mobile phones. This consists of development team, internal testing team and validation team. The product may also undergo test from external source (i.e. factory quality control) before delivering the product to the consumer. As the consumer satisfaction and the company image is the major concern for any product development company. The development team is into research of new applications for android mobile phone. Also other lower tier models are manufactured with lesser cost accordingly. Since the country adaptation is different for different countries. The quality has to be maintained as Chinese customers may finally end up in getting mobile hand set with applications supporting Indian languages. Intended software for correct user, functional and non functional working condition of the mobile hand set is tested manually, which may cause errors so automation plays a vital role for product development and release to factory [1].

The purpose of Development Life cycle model guidelines is to help projects deploy the standard

lifecycle model in the Software Department projects. The organization has defined the life cycle to be followed in the Software projects as below. This lifecycle has been named as “Software Development Lifecycle (SDLC)” and is found to be more prevalent among the projects in the Software department [2].

The following are the reasons for defining the Solution Department specific life cycle:

- Customer requirements: Customer would like to have intermediate milestones to check the product development progress. Hence all the projects are broken down into many milestones. The number of milestones in a project depends on the Complexity, size and project dependencies (availability of hardware etc).
- Lessons learnt: Prior to 2009, projects used to follow pure waterfall life cycle model for development. This created uncertainties in project till the product is delivered. As a risk mitigation strategy, organizations software group decided to use its own life cycle and the following model was defined.
- Evolving scope: Customer could evolve the scope of project after reviewing the progress during intermediate milestones. This provides more flexibility to customer and better service.

1.1. Software Development Lifecycle

A project is divided into multiple milestones. Each milestone needs to result in a working deliverable of Project and spanning between 3 to 4 months of duration.

The following is the architecture of milestone management of SDLC process model [3].

- M0: This basically constitute the high level system architecture and high level requirement analysis of the entire project. But, it would not involve any activities of deriving the detailed requirements of the subsequent milestones.
- M1, M2, M3...: This involves execution of all life cycle phases of SDLC (Requirement, Design, Coding, Unit Testing, Integration Testing, and System Testing). The detailed requirements of each of these milestones are derived as part of respective milestone only, by referring the high level architecture derived in M0.

Note: Milestones can be executed either sequentially or in parallel based on project requirement. Some of the combinations are listed below.

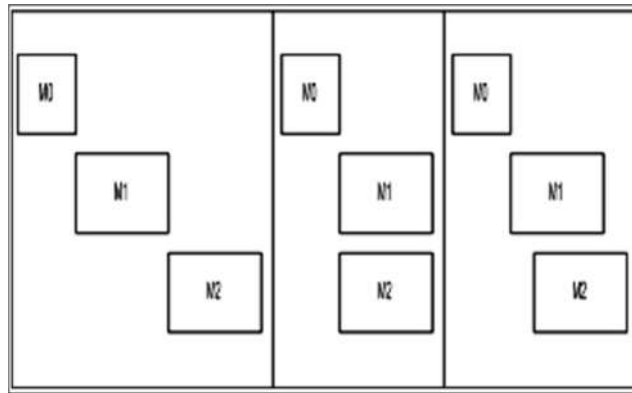


Fig 1: Milestone management for software development life cycle

The development life cycle is shown below

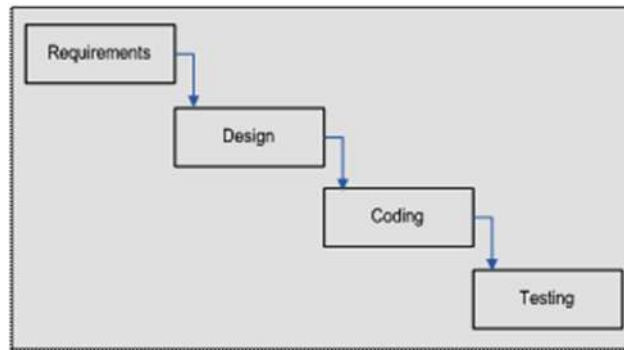


Fig 2: Software development life cycle

1.2. Phases

Phase and Process areas are distinct and should not be used interchangeably. The activities performed within the phases will be governed by Processes. For example, activities such as Unit Test planning and Unit Test performed within the Coding phase are governed by the Unit Test Process.

The following defines the scope of the various Phases:

- Requirements Phase: This phase consists of the following activities:
 - Requirements development
 - Requirements planning and allocation
 - Requirements reviews
 - Requirements rework
- Design Phase: This phase consists of the following activities:
 - Design
 - Design reviews
 - Design rework

- Coding Phase:
 - Coding, Code reviews, Code rework, Unit test planning
 - Unit test preparation including testcase generation
 - Unit testing
 - Integration planning, Integration, Integration test planning, Integration test preparation including test case generation, Integration testing
 - Unit testing and Integration testing related reviews and rework
- Testing Phase:
 - Test planning
 - Test preparation including test case generation
 - Acceptance test scenario generation
 - System testing and Acceptance testing related review and rework
 - System testing
 - Acceptance testing

The testing group has majorly two tasks, one is testing and the other is to release to downstream user. The team is provided with finite number of resources. But there's number of request from development team, which is variable and does not follow any pattern.

2. INTERNAL FLOW OF DATA IN ORGANIZATION

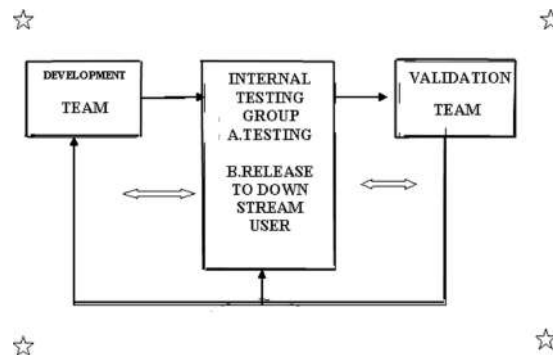


Fig 3: Internal organization structure

2.1. Task

The development team release information regarding the time taken for start and end of a model development. The internal testing group perform the various types of tests, here its sanity test and release the output to downstream users.

The validation team then validates the tests performed and certain control measures are specified as feedback to development leads.

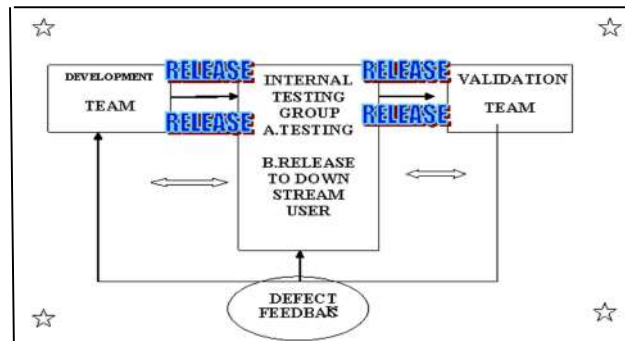


Fig 4: Internal flow of data in organization

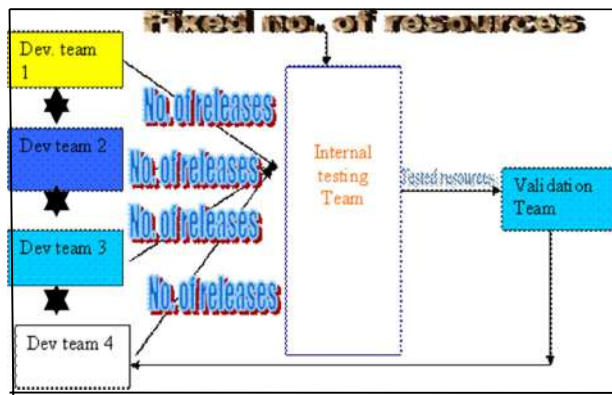


Fig 5: Real time release of data to various teams.

3. PROBLEM DEFINION AND ROOT CAUSES

The resources available from the development team will sent to the internal testing team as explained earlier , the risk involved in process is to be analyzed and the root cause for the risk is to be found. The Fig 7 explains that the maximum risk involved is due to resources which are not efficiently utilized by the internal organization.

Here we can clearly see that resource allocation has the maximum risk counts and is causing the problem.

The root cause for why the risk cont is high during resource allocation needs analyzed. After in-depth analysis of the organizations data release to internal testing team and verification team, human resources, manual testing of test cases, effort required for testing each test cases, effect of manual testing on productivity.

I found that two such cases which correspond to each other where in solving one problem may act as a solution to other problem.

Root cause 1: manual testing for test cases.

Root cause2: upload sequencing, server load and analysis of bottle necks in networks.

3.1 Problem statement with Details

3.1.1. Define the problem

- 20 releases per day.
- Scope is to pursue sanity, PRI, key press, Key Map.
- Covering all these things with existing resource is a challenging task

3.1.2. Identify where the problem is appearing

- IQM needs to test manually which requires a lot of effort.

3.1.3. Describe the size of the problem

- Sanity – 4 hours
- PRI- ½ hours per profile , 270 profile
- Key press – ½ hours per language, 35 language
- Key Map- 1 hour per model

3.1.4. Describe the impact the problem is having on the organization

- Delay in release to factory
- Increase VQM rejection

3.1.5. Preventive actions

Root cause 1: automation tool instead of using manual testing

Root cause 2: optimize the overall cycle time of server based on understanding of upload sequencing, server load and analysis of bottle necks in networks.

Here it's simple to understand as the test cases are automated the result of the test cases are known at a particular time

.Example: If the testing engineer runs the automation tool say at morning 9 am and expects the completion of all the test cases at 12 pm [4]. As the tool is automated result is fixed and time management is easy. Now resource planning is also simple as release the data related to model defects can be intimated to respective development leads.

The synchronization of release defect data to development leads was a major concern, now particular time is

been set to release the defect data .Example: If the result of testing was out at 12pm for model X , the test cases which have failed is noted and conveyed to the particular model development leads. So that the release of failure data related to model X is scheduled at 12:05pm with 5minutes delay time. Similarly this can be useful if the number of models released during a day is more than 10 to 15 for different country adaptation.

4. AUTOMATION TOOL

As of now management tools such as minitab, similar, matlab and excel sheets were used to analyze the data but now the real work is to develop an automation tool[11]. Based on the user requirement the automation tool was developed where in excel sheet was the input and basic languages such as C and C# was used to write the source code.

Example: TestMain() function for testing phonebook in amobile phone[7].

```
void TestMain()
{
    wchar_t PhonenumP[50];wchar_t ActValue[15];
    char TempValue[15];
    //int i = 0,j = 0;
    //wchar_t**Menu_Item,*Case_Item,value[15],MenuName[100],
    PreMenuName[100],*FullScreen,*KeymapPath, *ImgPath;
    //wchar_t *SheetName,*ExlValue,*Test,*Flag,*ExlValue_Temp;
    //wchar_t *checkbox=(wchar_t*) malloc (sizeof(wchar_t)*25);

    //KeymapPath = DEV1.GetMacroValue(8);
    ExcelWorkbook*wb=DEV1.ExcelOpen(KeymapPath,L"Sanity");
    wb->SetActiveWorksheet(L"Sanity");
    wchar_t **MenuItem= (wchar_t**)malloc(sizeof(wchar_t)*30);

    wchar_t**MenuItem_Number=(wchar_t**)malloc(sizeof(wchar_t)*30);    int    **regionValue=
    (int**)malloc(sizeof(int)*30); memset(checkbox, NULL, sizeof(wchar_t)*25);

    for(i=BASICTESTCASE143;i<=BASICTESTCASE158;i++)
    {
        sprintf(TempValue, "J%d", i); mbstowcs(ActValue,TempValue,strlen(TempValue-1));
        checkbox = wb->GetCellValue(ActValue);

        if((i == BASICTESTCASE143) && (!wcscmp(checkbox, L"True")))

    {
        Basic_Test_Case_143(wb);          DEV1.KeyPressEx(L"E",10,1000);
        DEV2.KeyPressEx(L"E",10,1000);
    }
}
```

```

else if((i == BASICTESTCASE143a) && (!wscmp(checkbox, L"True")))
{
    Basic_Test_Case_143a(wb);
    DEV1.KeyPressEx(L"E",10,1000);
    DEV2.KeyPressEx(L"E",10,1000);
}
}

void Basic_Test_Case_143(ExcelWorkbook *wb)
{
    wchar_t PhonenumP[50]; wchar_t *p,*KeymapPath;
    p=(wchar_t*) malloc(sizeof(wchar_t)*50); char TempValue[15],column;
    wchar_t ActValue[15],result1[25]; wchar_t
    *WaitValue,*ExIValue,*ExIValue_Region,*ExIValue_Region1,*ExIValue_Region2;
    int
        rowNum
        =
    BASICTESTCASE143,ret,countvar,j,count,ExIValue_Sleep;

    SYS_Debug("\n TC,To receive a SMS and check"); printf("\n TC,To Add a Contact
    in Phone Book"); DEV1.KeyPressEx(L"E",10,1000);

    wb->SetActiveWorksheet(L"Sanity");
    DEV1.TouchClickAction(wb,L"Sanity",rowNum,'K',1);

    KeymapPath = DEV1.GetMacroValue(8);
    wb-
    >SetActiveWorksheet(L"Sanity");

    wscpy(PhonenumP,DEV2.PhoneNumber);
    p=PhonenumP;
    printf("%S\n",p);
    printf("%c\n",p[0]);

    //
    wb
    =
    DEV1.ExcelOpen(KeymapPath,L"MSGDIALER");
    wb-
    >SetActiveWorksheet(L"MSGDIALER");
    writingPhoneNumberEditor(wb,p);

    wb->SetActiveWorksheet(L"Sanity");
    DEV1.TouchClickAction(wb,L"Sanity",rowNum,'L',1);

```



```
//      wb->SetActiveWorksheet(L"Sanity");
DEV1.CommonAction(wb,rowNumber);
}

void Basic_Test_Case_143a(ExcelWorkbook *wb)
{
    int rowNumber = BASICTESTCASE143a;

    SYS_Debug("\n TC,To receive a SMS and check"); printf("\n TC,To Add a Contact in
    Phone Book"); DEV1.KeyPressEx(L"E",10,1000);

    wb->SetActiveWorksheet(L"Sanity"); DEV1.TouchClickAction(wb,L"Sanity",rowNumber,'K',1);

    wb->SetActiveWorksheet(L"Sanity"); DEV1.CommonAction(wb,rowNumber);

}
}
```

[illegible]

Fig 6: Input for automation tool- excel sheet

Fig 6 shows the excel sheet which is used as an input device where in the mobile screen images are captured and verified. The automation tool also provides inbuilt functionalities to set up new navigations to different mobile phones.

5. DEPLOYMENT

5.1 Raw data

The data collected from March to October from the respective development leads and with the help of similar software and also manually analyzed the given metrics, also prepared a line chart for number of models-months taken for developing particular models[6].

The development of the models and duration for development is also verified and for better understanding line charts were created as shown in Fig 8.

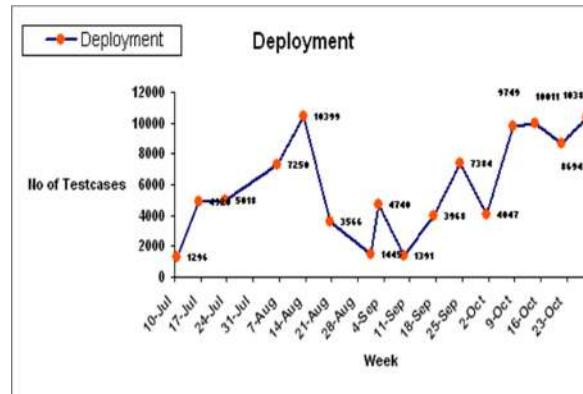


Fig 7: Deployment of No. of test cases-week

The Deployment of test cases data were also collected from model leads and analyzed the variance in the particular data per month. Sanity test which is for functional application such as key press, key map were collected and the time taken for each test conducted are shown below in Table 1. As we know that the functioning of keys in a mobile phone is a major criterion so the testing team usually conducts such tests frequently[10]. Here the test leads conduct sanity test and the time management system which is an internal system in the organization calculate the time taken for each test and also time taken for each activity is also noted such as key map, key press and any special applications such as word dictionary. The Table 1 clearly shows the test conducted and the time taken for each month. After the further analysis of data for future upcoming months

MANUAL V/S AUTOMATED.

Nowadays manually executing the processes takes more time and the human resources required for the verification of test cases is humongous. Even though the internal testing team works in different shifts, manual deployment of test cases takes more time the remedy for such bottle necks is to automate a tool which helps the testing of functional like sanity testing and non-functional like setting alarm, deleting messages, stress checking the contact capacity which may be limited to 250 contacts only. Here the automation is essential in non-functional test cases where the test engineer can test certain applications such as stress checking any application for more the one day, even if he's not in working place he can run the automation tool which helps him to stress check that particular application and if there is any critical issue related to test cases then the development lead can be intimated which helps the

development leads to know the error well in advance and control measures can be taken to overcome this problems. But in case of manual testing for functional applications like the sanity test companies prefer manual testing itself as knowledge of different domain is essential for a tester which cannot be updated in an automation tool. This untouched area wherein manual testing was essential, an automation tool is developed which reduces the burden of the tester and also manual vigilance is applicable. Here we use resource planning and cycle time reduction in testing group. The types of test performed are Sanity Testing or detailed testing...etc. is a brief test of major functional elements of a piece of software to determine if it's basically operational. Sanity testing is used to verify that the software build is ready for System Testing (or more extensive testing). The tests target the most frequently used functional areas, key functions & newly implemented functions for the particular build version.

If an application does not pass the Sanity Test, then further testing activities are suspended. The conditions which get checked in this test is general, like existence of application, loads without any problem, does not terminate abruptly, connects to correct database, all menu options are displayed correctly and the start form / page / window is loaded correctly.

The comparison of manual deployment of test cases like the functional and non functional test cases which is compared with the automated tool evaluated test cases. Here we can see a drastic reduction in time for single test case e.g. - if a manually tested test case takes one hour then the automated tool tested test case takes three minutes, it's an astonishing 57 minutes difference between the manual and automated test case.

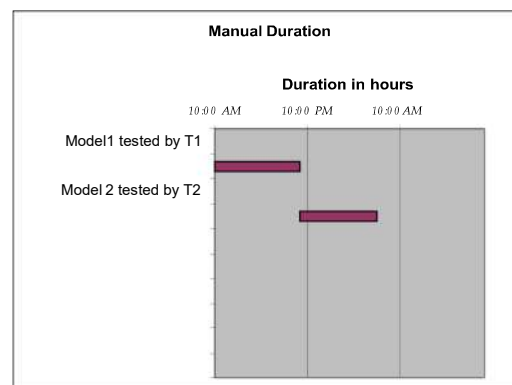


Fig 8: Gantt chart for manual testing

Parameter	Manual Execution (What IF Analysis)	Automation
No of Cases	94295 test cases	94295 test cases
Test efficiency	157.94	283.12
No of engineer	4	4

Man month	7months	4month
Effort required	27 man month	15 man month
cost	27000	15000
Saving		12000 USD

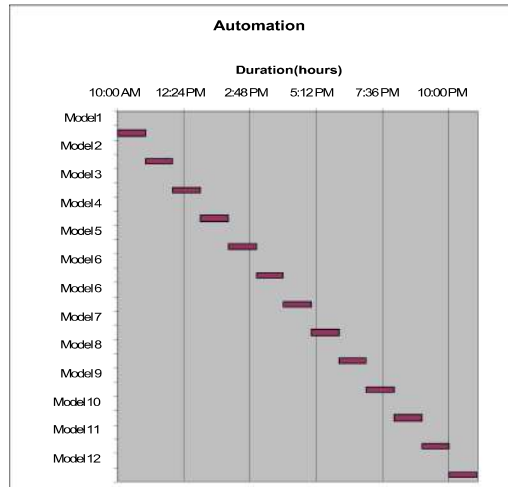
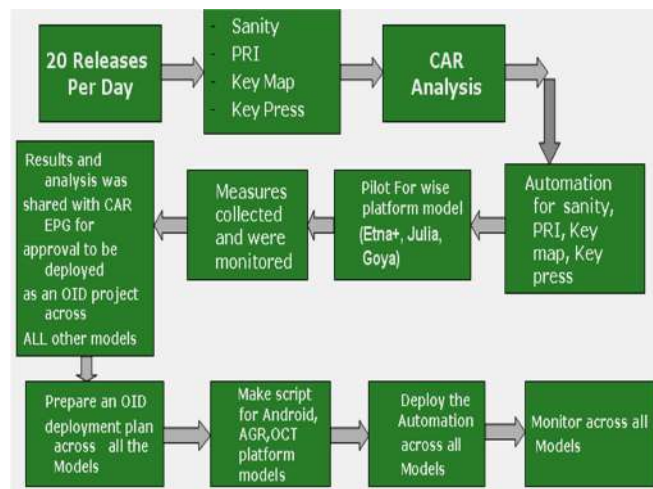


Fig 9: Gantt chart for Automation



DIFFICULTIES FACED DURING MANUALDEPLOYMENT

- Available resources may be different
- Speed of release is reduced as manually verifying the test cases may take some.
- Volume of the testcases constant and verification is bulk work.

- *Late night releases which causes leads to stay for more than expected time.*
- *Cost for both human resources and release .*
- *Extra computer is required for manual testing[8].*

6. BENEFITS OF AUTOMATION

- *Extra computer for testing perpose is not required*
- *Automation may increase the speed of releases to downstream users.*
- *Voluminous jobs can be distributed.*
- *Cost saving.*
- *Accuracy is maintained.*
- *Errors caused during manyual testing is reduced*
- *Test engineers have their work cutoff.*
- *Stress testing is possible as the test engineers may run the automation tool for several days .*

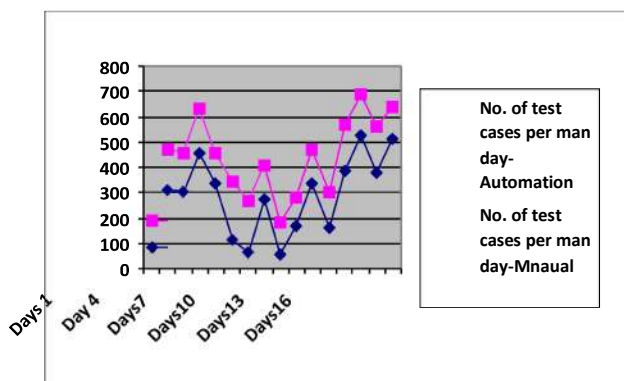


Fig 10: Increase in productivity per due to automation Fig 11: Cost Benefit Estimate

5.

7. CYCLE TIME REDUCTION

Cycle time is the time required to complete a given process. The cycle time required to process a customer order might start with the customer phone call and end with the order being shipped in this example. The overall process is made up of many sub-processes such as order entry, assembly, inspection, packaging, and shipping. The cumulative cycle time of all of the sub-processes in your operation determines when you can promise product to your customer.

Benefits of cycle time reduction?

- Reduced costs
- Streamlined processes
- Improved communications

- Improved on-time delivery
- Improved productivity

How can Automation tool help implement a Cycle Time Reduction plan?

Automation tool works with you and your team to understand your business and its processes[9]. Together, we map out your company processes, investigating new methods and identifying opportunities for reducing non-value added activity. Once this is complete, the field engineer helps you create and implement a plan for cycle time reduction.

As shown in Fig.14 if the defects are verified well in advance in loop 1 itself the errors carried on to further processes is reduced. The development team leads and also the test leads are intimated with the errors so that the leads can take up control measures to overcome errors.

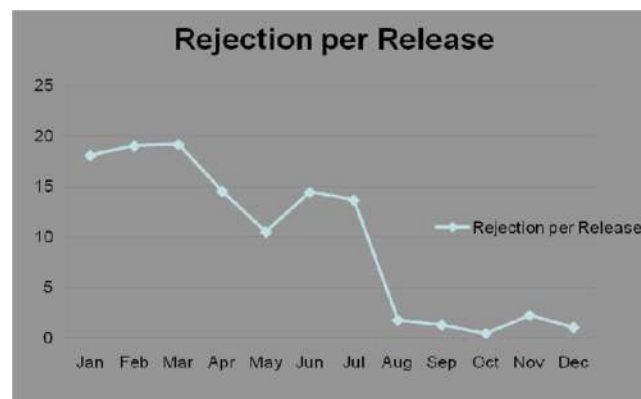


Figure 12: Decrease in rejections per release decrease cycle time

The Fig 12 shows the decrease in rejections per release due to automation, this saves time as more rejection may under go repeated testing. The decrease in rejection per release clearly shows that cycle time is reduced and the product is delivered to the customer or the sales intermediate channels in time.

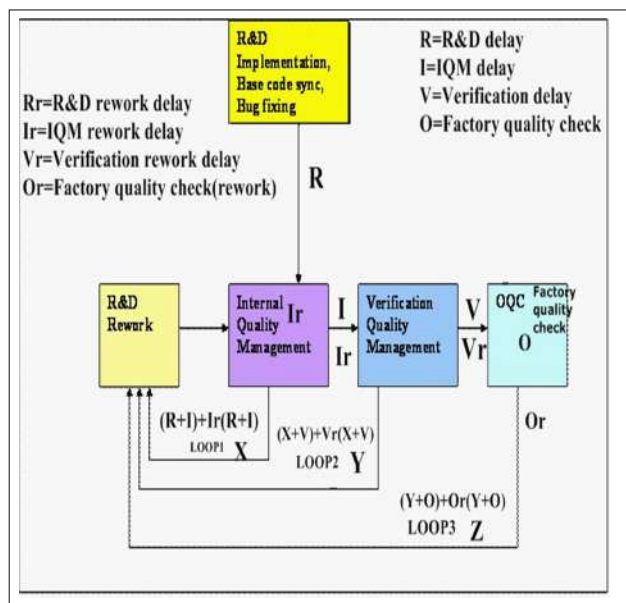


Fig 13: Calculation of cycle time

The above figure explains about the time taken for rework if the defects are carried on to next stages .Its obvious that the cost and human resources required for the rework or the defect correction is humongous .The calculation part is explained below .

$$(R+I)+Ir(R+I) \rightarrow (X)$$

$$(X+V)+Vr(X+V) \rightarrow (Y)$$

$$(Y+O)+Or(Y+O) \rightarrow (Z)$$

Rework	Internal quality management	Verification Quality Management	Factory Quality management
Ir	1	0	0
Vr	0	1	0
Or	0	0	1

Table 1:Banaries for calculating cycle time.

6. UPLOAD SEQUENCING AND SERVER LOAD.

Finally, the next task after automation tool for test cases is to release to downstream users, which takes a lot of bandwidth for testing group to upload the document and source code. Optimization of server based on understanding of uploadsequencing and server load [11]. To meet the current load which will save several million dollars as the server cost is high.

8. CONCLUSION

In this competitive world customer satisfaction is main concern for any company. Hence delivery of the product rightin time with quality of the product also maintained certainly takes some pain. As we know that “out of sight, out of mind”,the errors carried on to factory and then to distributor, finally to customer leave a bad impact on the reputation of the company. In the FIG.13 if the resource planning and cycle time reduction is efficiently carried on during loop 1 itself then any critical error caused while development leads are negotiated then and there itself ,which saves time and cost incurred for retesting the product[12]. An automation toolcan help human resource and extra computer required for testing, voluminous jobs can be distributed, speed of testing is increased and finally overall cost is reduced .The second issue which helps in scheduling the release to downstream users which causes server down scenario which can be reduced, the test leads as well as the development leads will know when to release the data , also to predict and optimize overall cycle time of server based on understanding of uploadsequencing ,server load and analysis of bottle necks in network. To meet the current load a new server has to be installed which is costly, optimization of cycle time will handle this issue efficiently .

REFERENCES

- [1] Dr. Bhargav Gangadhara, "Optimizing Cloud - Based Manufacturing: A Study on Service and Development Models", International Journal of Science and Research (IJSR), Volume 12 Issue 6, June 2023, pp. 2487-2491, <https://www.ijsr.net/getabstract.php?paperid=SR23626155823>
- [2] Dr. Bhargav Gangadhara, "Optimize Utility in Computing-Based Manufacturing Systems Using Service Models and Development Models", Presented and published at International Journal of Computer Science and Information Security, IJCSIS Editorial Board, Vol -14, Page 5.
- [3] Dr. Bhargav Gangadhara, "Optimize utility in cloud-Based manufacturing systems using service models and development models", Presented and published at International Journal of Innovative Research in Advanced Engineering. IJIRAE Editorial Board, August 11 - 12, 2016.
- [4] W.Humphrey. "Managing the Software Process." Addison-wesley, 1989.
- [5] B.W. Boehm. "Software Engineering economics." Prentice Hall, 1981.
- [6] D.C. Montgomery. "Introduction to statistical Quality Control," third edition. John Wiley and Sons, 1996.
- [7] A.J. Albrecht and J.R. Gaffney. "Software function, source lines of code and development efforts prediction: a software science validation." IEEE Transaction on Software Engineering, 9(6):639-648, 1983.
- [8] J.D. Musa, A. Iannino, and K. Okumoto. "Software Reliability – measurement, Prediction, Application." McGraw Hill, 1987.
- [9] R.N. Charette. "Large-scale project management is risk management", IEEE Software, pp. 110-117, July 1996.
- [10] R.G. Ebenau and S.H. Strauss. "Software inspection process." McGraw Hill, 1993.
- [11] D.H. Kitson and S.M. Masters. "An analysis of SEI software process assessment results: 1987-1991. Proceedings of the 15th international conference on software Engineering, pp. 68-77, 1993.
- [12] D.P. Youll. "Marketing software Development Visible—Effective Project control." John Wiley and sons, 1990.