IJETCR Open Access

ISSN: 2348 - 2117

International Journal of Engineering Technology and Computer Research (IJETCR)
Available Online at www.ijetcr.org
Volume 3; Issue 2; Page No. 102-106

A Survey on Incremental Software Development Life Cycle Model

Ankit Thakur[1], Deepti Singh[2], Harish Chandra Maurya[3]

[1] *M.Tech (C.S) (Scholar, Bhagwant University, Ajmer, India)*

[2] *M.Tech (C.S) (Scholar, Bhagwant University, Ajmer, India)*

[3] *M.Tech (C.S) (Assistant Professor, Bhagwant University, Ajmer, India)*

ABSTRACT

In this current era of software development, a large number of life cycle models are available for the systematic development of computer software and projects. SDLC models give a theoretical guide line regarding development of the software. This paper focused on the survey of incremental software design life cycle. The incremental model is a method of software development where the product is designed, implemented and tested incrementally until the product is finished. It involves both development and maintenance. The product is defined as finished when it satisfies all of its requirements. This model combines the elements of the waterfall model with the iterative philosophy of prototyping. Incremental model is an evolution of waterfall model. The product is designed, implemented, integrated and tested as a series of incremental builds. It is a popular model software evolution used many commercial software companies and system vendor. Incremental model is the advancement of the waterfall model. The phases of waterfall model are employed in such a manner that the result of any of the increment is used back as the input for the next increment. Thus with each increment there are some client's feedback that is used in getting the next incremental product. Thus with each ongoing increment the functionality of the core product gets enhanced. The series of releases is referred to as —increments , with each increment providing more functionality to the customers. After the first increment, a core product is delivered, which can already be used by the customer. Based on customer feedback, a plan is developed for the next increments, and modifications are made accordingly. This process continues, with increments being delivered until the complete product is delivered.

Key Words: Incremental model, SDLC, Waterfall model, Software engineering and Software

1. INTRODUCTION

Incremental model is the best model to be used for both large and small project rather than any other models. This is because incremental model composes of the Waterfall and Prototyping model. Waterfall model is mainly targeting the small project. The other weakness of the waterfall model is that it does not require the designer to step back for correcting errors. Therefore the incremental model combines of the two models hence it will tackle both small and big projects. Incremental model also handle errors pretty well as they can be easily identified due to the requirements being broken down into smaller units. This make the incremental the best commercial method to be used as for iterative and linear models only allow errors to be recognized until acceptance testing, when it is probably too late to correct the errors. With incremental development lifecycle models, risk of developing the wrong thing is reduced “by breaking the project into a series of small subprojects [increments]. The total scope of work is decomposed to smaller chunks of work, increments, based on the risks, architecture and/or the requirements. In incremental models, as in sequential models, the overall requirements of the final system or product are known at the start of the development. In incremental models however a limited set of requirements is allocated to each increment and with each successive (internal) release more requirements are addressed until the final (external) release satisfies all requirements. One risk with the incremental approach is that the first releases addresses such a limited set of requirements that the customer could be dissatisfied, one opportunity on the other hand is that wrong or missing requirements can be corrected in time.

The paper is organized as follows. In the Section -2, Incremental model approaches are discussed, Section -3 summarizes incremental model phases, Section-4 includes the advantages and disadvantages of model and final discussion is concluded in Section-5.

Corresponding author: Ankit Thakur

2. INCREMENTAL MODEL APPROACH

In incremental model the whole requirement is divided into various builds. Multiple development cycles take place here, making the life cycle a "multi-waterfall" cycle. Cycles are divided up into smaller, more easily managed modules. Each module passes through the requirements, design, implementation and testing phases. A working version of software is produced during the first module, so you have working software early on during the software life cycle. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is achieved.

Figure 1: Example for incremental model

In the diagram above when we work incrementally we are adding piece by piece but expect that each piece is fully finished. Thus keep on adding the pieces until it's complete. As in the image above a person has thought of the application. Then he started building it and in the first iteration the first module of the application or product is totally ready and can be demo to the customers. Likewise in the second iteration the other module is ready and integrated with the first module. Similarly, in the third iteration the whole product is ready and integrated. Hence, the product got ready step by step. Consider an example where a bank wants to develop software to automate the banking process for insurance services, personal banking, and home and automobile loans. The bank wants the automation of personal banking system immediately because it will enhance the customer services. You can implement the incremental approach to develop the banking software. In the first increment, you can implement the personal banking feature and deliver it to the customer. In the later increments, you can implement the insurance services, home loans, and automobile loans features of the bank. The regulation of the incremental representation is that it is applicable only to large and bulky applications. This is because it is hard to break it into small serviceable increments.

2.1 When to use the Incremental model:

- This model can be used when the requirements of the complete system are clearly defined and understood.
- Major requirements must be defined; however, some details can evolve with time.
- There is a need to get a product to the market early.
- A new technology is being used
- Resources with needed skill set are not available
- There are some high risk features and goals.

3. INCREMENTAL MODEL PHASES

Each increment includes three phases: design, implementation and analysis. During the design phase of the first increment, the functionality with topmost priority from the project activity list is selected and the design is prepared. In the implementation phase, the design is implemented and tested. In the analysis phase, the functional capability of the partially developed product is analyzed. The development process is repeated until all the functions of the project are implemented.

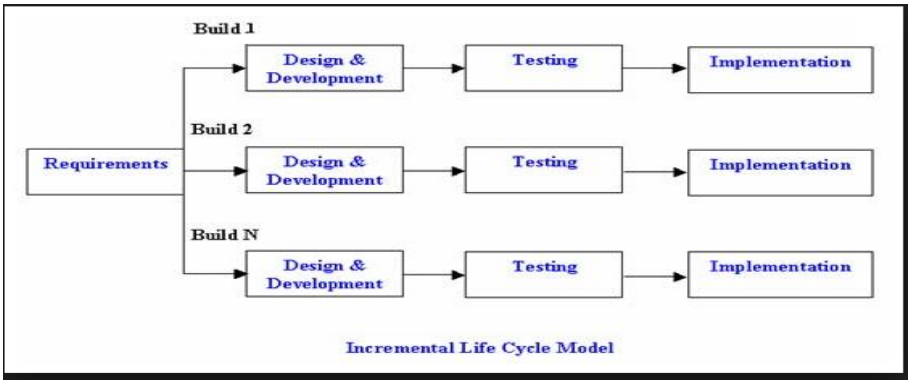

FIGURE 2: PHASES OF INCREMENTAL MODEL

3.1 ANALYSIS

In the analysis stage, before we can think of solving a problem it's very significant to study the current system before you can start working on major changes. When software is developed the developers needs to understand the problem for which the software is to solve. Development team should meet the client to study their system. For a program to be successful the analyst must understand the features, functions, behavior and performance of the system. The problem could be to increase the speed of the existing or to automate the system or simply to develop a new system. Large software that has many features requires the developers' good skills and experience to take proper decisions. The main task of the analysis stage is to identify the expected needs from the system. Analysis in the incremental model focuses on find out the influences that makes the organization to develop a new system or modify the existing system. There are some questions that need to be asked during analysis. These are as follows:

3.1.1 Is there really a problem?

This question helps to investigate and know whether it's necessary to start a new project or modify the existing system. Imagine a scenario where a manager wants to change the organizations selling operations by spending thousands of money and many hours to implement an online shopping system. After it's launched, the manager discovered that he wasted a lot time and wasted hundreds and thousands money because no-one actually uses it because they were happy with the old system.

3.1.2 If there is a problem, is it worth fixing?

Let's say it is now discovered that there is a problem, so what need to be done now is to consider a "cost/benefit" analysis.

Some of the changes which can influence the organization change the current system may be of the following:

- Change in technology: changes of the running system can be done due to change in technology. A system can be changed from a manual to automatic. As technology change rapidly the need for organization to change their operation for better services. New machinery or technology may enhance efficiency in the use of labour and material. It is clearly important to know whether the improvements made may affect the organization positively.
- A company can buy a machine in order to increase its productivity. The company needs to make a research concerning the type of the machine they want to best satisfy their needs. If it happens that they don't make a good research they can spend money on buying a machine which will not meet their needs. Suppose that the machine they have bought on the 1st June cost million Pula and the following day that is on the 2nd June they fund that the type of machine they want cost P500000 which makes twice the productivity of the first machine. This shows that the company has made a loss "sunk cost".

3.1.3Consider an example below:

Think of an Analysis phase like a visit to an IT (information technology) technician. You will be pretty concerned if you told the technician that your computer doesn't work and immediately jump on removing different parts before even examining it or ask you any questions. Probably such kind of behavior may raise more problems than it solves. So technicians analyze problems first - observe, question and test before beginning to remove computer parts.So this example above is similar to the SDLC models where the problem solvers have to study the system they want to change in an organization. Before they can decide on what has to be done. They will have to study and understand the organization's operation, its context, its strengths and weakness. After all this is done, now it will be time to decide how to start improving it.

3.2 DESIGN

The design phase actually describes how the solution will work, once the structure of the software is confirmed by the management, the architecture of the program is now created. A work will be left for the programmers to choose the ways of developing the program. The top-down approach is used to develop and link major programs. The bottom-up approach is then used to combine small units with the main program. The steps involved in the development are drawn down and will be followed to develop the program. The programmer needs to decide the best model that is suitable for the program. Incremental design has a new process of processing design in a structured way that will allow the user to reuse the other part of the design which is unchanged. This can save time for the designer or the processing of each iterative. Since the increment model is divided into functional unit the result that is done get from the first design of the first functional unit is used as the input of the next design of the second functional unit. Incremental Design flows for new and existing designs. Before running any Incremental Design flow, the design should meet the requirements without using Incremental Design. For Incremental Design to significantly reduce runtime and maintain performance for unchanged portions of the design, Incremental provides few benefits to the design.

3.2.1 Saves time

Design of the incremental model saves time as the design of the first phase can be modified or part of the design can be reused. The other thing that saves time is the use

of the output of the first functional unit to the next unit. This is being done with the help of feedbacks from the customers.

3.2.2 Easy to test

It is easy to test since the project is divided into functions. Errors are identified and managed easily as the project is divided into smaller units.

1.3 IMPLEMENTATION

Once testing is finished and the software is proven good for implementation, it is released to the organization. It's expected that developers will face serious challenges of fixing different bugs as they are discovered one by one by different users. The main difference of the implementation stage to the testing is the amount of bugs expected. System installation is the process whereby the newly developed system is installed. Both old and new software are used together for about one or two months until it is confirmed that the newly developed system is working as required without any bugs. After it has been confirmed that it is working properly the old system is removed.

3.4 TESTING

Testing is one of the important stages of the SDLC lifecycle. The main task is to detect errors in the software. A quality produced work needs to be tested regularly. Testing is an activity that needs to be done throughout the life cycle. It is also used as a major source of feedback. Testing should not be done at the end of the software as it can be dangerous. Rather it has to be done timely throughout the stage. . There are number of test that are used during testing phase.

- Unit testing
- Regression testing
- Stress testing
- Functional testing
- Integration testing

Programs in the incremental model are coded in various functional units, these subjects to separate and detailed test. After the first functional unit get completed; it taken to the customer to have a review on it. Feedback provided by the customer concerning the unit is used to improve/modified to the next unit (output of the first unit is used as the input of the next functional unit. This process is going to take place until customers are satisfied. The reasons why this system is tested is to check if the interfaces between functional units work together (this is known as integration testing), check if the system works on the specified platform and with the required volume of data (also known as volume testing), and checks if the system meet the users requirements (it is known as acceptance/ beta testing). Some of the test that can be used in the incremental model:

3.4.1 Unit Testing

This is where the testing of each individual components are done to check if there are working. The programs that make up the system are tested here. It tests modules separately and this allows tester to discover errors in coding and logic inside a particular module.

3.4.2 Regression Testing

It is done to test again the past tested segments of the system to make sure that they are work correctly after a change was made in another part of the system. To make sure that operational performance does not go down because of the changes and they are no new problems introduced, the system should be tested thoroughly.

3.5 MAINTENANCE

This stage of SDLC is used to correct/maintain the software if gives the client some problems. Post delivery maintenance is done after the software is delivered to the user. Maintenance can be done for different purposes. Corrective maintenance: is a type of maintenance which is done when a fault has occurred either to the coding part, design or documentation. Software problems may be due to system failure or discovery of software errors. Perfective maintenance: as the word denotes the meaning, it is a type of maintenance that is done to change the functionality of the software. A change can be done at the coding part of the software. A change can be done to increase the performance of the software or even its functionality. Adaptive maintenance: the word adaptive means to get used to something or the environment in which the product operate. Adaptive maintenance is done when the product is not working perfectly according to the user expectation. Software adaptive can be done when the software is not portable with the type of hardware it is installed in.

4. ADVANTAGES AND DISADVANTAGES

4.1 Advantages of Incremental model:

- Generates working software quickly and early during the software life cycle.
- This model is more flexible – less costly to change scope and requirements.
- It is easier to test and debug during a smaller iteration.
- In this model customer can respond to each built.
- Lowers initial delivery cost.
- Easier to manage risk because risky pieces are identified and handled during it'd iteration.

4.2 Disadvantages of Incremental model:

- Needs good planning and design.
- Needs a clear and complete definition of the whole system before it can be broken down and built incrementally.

- Total cost is higher than waterfall.

5. CONCLUSION

This paper concludes that the incremental model is the best model in commercial projects because it accumulates all sorts of software development whether small or large. The model satisfy the clients and benefit both parties i.e. developers and clients because the product will be developed and delivered well in time compared to a situation where a programmer will be coding to solve a problem without analyzing, designing, and testing.

REFERENCES

1. Royce, W. W. (1970). Managing the development of large software systems: concepts and techniques, Proceedings, Wescon 1970.
2. Blake, Jeffery, "Software through Pictures vs. Paradigm Plus," Advanced Systems Magazine, June 1994, p. 84
3. D. O'Neill, "The Management of Software Engineering," IBM Systems J., vol. 19, no. 4, 1980, pp. 421-431
4. H. Mills, "Principles of Software Engineering," IBM Systems J., vol. 19, no. 4, 1980, pp. 289-295.
5. Nabil Mohammed Ali Munassar Ali and Govardhan A (2010) "A Comparison between Five Models of Software Engineering" International Journal of Computer Science, Vol. 7(5), 98-100.
6. T Bhuvaneswari and Prabaharan S.(2013) "A Survey on Software development life cycle model", Journal of Computer Science and Information Technology, Vol2 (5), 263-265.
7. Eduardo Malaga Chicano, "A comparative study if iterative protyping vs waterfall model processed by small and medium sized projects by system engineering", 1996.
8. Craig Larman and Victor R. Basili,"Iterative and Incremental Development: A Brief History". IEEE Computer (IEEE Computer Society) 36 (6): 47–56. Doi:10.1109/MC.2003.1204375. ISSN 0018-9162. Retrieved 2009-01-10.], June 2003.
9. Ernest Mnkandla, "About Software Engineering Frameworks and Methodologies", IEEE AFRICON 2009.
10. Sanjana Taya, Shaveta Gupta, "Comparative Analysis of Software Development Life Cycle Models"
11. Mohamed Sami Abd EI-Satar" Software Development Life Cycle Models and Methodologies", 2012
12. Kevin Forsberg and Harold Mooz, "The Relationship of System Engineering to the Project Cycle," in Proceedings of the First Annual Symposium of National Council on System Engineering, October 1991: 57–65.
13. Royce, Winston, "Managing the Development of Large Software Systems", Proceedings of IEEE WESCON 26, 1970
14. Jeffrey A. Livermore Walsh College, "Factors that Impact Implementing an Agile Software Development Methodology".
15. Lillian Doric and Elias Niemelae "A Survey on Software Architecture Analysis Methods", IEEE Transaction on Software engineering, VOL. 28, NO. 7, JULY 2002.
16. M, Hue, 1. Varner, L. Zhu, and M, A. Babar, "Software quality and Agile Methods," Pros,28th Annual International Computer Software and Applications Conference (COMPSAC'04), pp., 520-525,2004
17. Steve Easterbrook, "Software Lifecycles", University of Toronto Department of Computer Science, 2001.
18. JJ Kuhl, "Project Lifecycle Models: How They Differ and When to Use Them" 2002.
19. Harish Rohil and Syan Manisha(2012) "Ananlysis of Agile and Traditional Approach for Software development", International Journal of Latest Trends in Engineering and Technology ,Vol. 1(4),1-3
20. Fowler, M., "Put Your Process on a Diet", Software Development",2000 services, page 45, Washington, DC, USA, 2007.