Improving Resource and Manpower Allocation Using Enhanced Software Development Model for Efficient Generation of Software

[1]Nagajan Tarkhala, [2]Prof. Chintan N. Kanani

[1]Student, Master of Engineering, Darshan institute of Engineering & Technology, Rajkot, India.
[2]Lecturer, Darshan institute of Engineering & Technology, Rajkot, India.

***Abstract* - At Present all major routine work is being converted electronically, so there is a certain need about considering Software Engineering as an emerging and growing domain. It is not practically or physically possible that flow of software generation can be mapped with its mathematical output. In present all software systems are imperfect as there is always probation in development without any fixed certainty. According SDLC each and every model has the advantage and drawbacks. So in this research we have to calculate the performance of each model on behalf of some important features. This paper categorizes and examines a few methods for relating or modeling how software systems are developed. In this paper, we are going to compare various software development models using various parameters like resource, manpower, quality and profit ratio to show the features and defects of each model. The major concentration of the paper is to provide software development model with less efforts and less resources with high quality output and for that we are going to generate new SDLC model that can be useful for any organization in any situations.**

***Index Terms* - Software Development Process, SDLC, phase of SDLC, Waterfall, Iterative.**

I. Introduction

Everyone broadly accept the importance of computer in our life, especially during the present time. In fact, computer has become indispensible in today's life as it is used in many fields of life such as industry, medicine, commerce, education and even agriculture [1]. A software development process, also known as a software development life cycle (SDLC), is a structure imposed on the development of a software product [2]. Different processes and methodologies have been urbanized over the last few decades to improve software class. However, it is broadly approved that no single approach that will prevent project overruns and failures in all cases. A software life cycle model is either a descriptive or prescriptive characterization of how software is or should be developed. A descriptive model shows the narration of how a particular software system was developed [3] [1]. Normally, it is easier and more common to eloquent a prescriptive life cycle model for how software systems should be developed. This is possible since most such models are perceptive or well reasoned. This means that many individual details that describe how a software system is built in practice can be ignored, generalized, or deferred for later consideration [4][5]. Software development teams, taking into account its goals and the scale of a particular project, and have a number of well-established software development models to choose from. Therefore, even though there are number of models, each Software Development Company adopts the best-suited model, which facilitates the software development process and boosts the productivity of its team members [6]. So, according to the requirement, team selects approach and same as we are going to suggest that team should use more than one approach or combined approach for developing the same software. So that as a result they need not to compare all categories of their requirement to generate model [7] [10] [18] [20].

II. Software Development Models in Brief

A Programming process model is a conceptual illustration to explain the process from a particular viewpoint. There are numbers of general models for software processes, like: Waterfall model, Iterative development, Prototyping etc. [1] [3] [5] [7] [14] [15].

2.1 Waterfall Model

The waterfall model is the classical model of software engineering. This model is one of the oldest models and is widely used in government projects and in many major companies. As this model emphasizes planning in early stages, it ensures design flaws before they develop. In addition, its intensive document and planning make it work well for projects in which quality control is a major concern. [1][14]

2.2 Iterative Model

The troubles with the Waterfall Model formed a claim for a new process of developing systems which could give faster results, require less up-front information, and propose larger flexibility. With Iterative development, the projects are divided into minute sections. This makes the team to display results earlier on in the process and get important feedback from users. [2][4]

2.3 Prototyping

The novel purpose of a prototype is to allow users of the software to evaluate developers' ideas for the plan of the ultimate product by really trying them out, rather than having to take to mean and estimate the design based on metaphors. [3]

These are the three different models that are broadly used in IT Industry but they have drawbacks as waterfall is the basic model for beginners and iterative approach is dynamic but takes more time.

III. Comparison of various params

In Previous section we discussed about various approaches to develop software with many options and parameters. Now in this section we need to compare those papers with respect to almost all parameters to come to know that which model is good to use in all cases or what improvement should be there to work with the models as a part of research. We are going to combine various approaches to make an efficient model that is suitable for all kinds of requirements of software development. Here comparison is based on many parameters but the highlighted parameters are risk analysis, man power, resource consumption etc [1] [2] [4] [7] [8].

Table 1: Comparison of s/w models with various params

Features	**WF**	**IT**	**Pro**
Requirement	B	B	FC
Understanding	WU	NU	NU
Cost	L	L	A
Time	H	H	H
Risk	H	M	A
Errors	L	H	M
Man power	M	M	A
Resources	H	H	H
Reusability	N	N	Y
Complexity	L	L	L
Success Ratio	L	L	M
Merging Phases	N	N	N
Flexibility	N	Y	N
Changes	N	N	Y
Size of Project	S	S	W
Expertise	N	N	Y
Cost Control	N	N	Y
Resource Control	N	N	Y
Manpower Control	N	N	N

Terms for Comparison table:

Y-Yes N-No	**L**-Less	**H**-High
B-At Beginning	**A-** Average	**M**-Moderate
WF – Waterfall	**IT**-Iterative	**SP**-Spiral
Pro-Prototyping	**NU**- not well understood	**WU**-Well Understood

IV. Proposed Work

In previous sections, we compared various software development models with respect to various 20 parameters. The general two approaches are now a days used in industry is waterfall and iterative for developing software.

4.1 New approach for software development

Our goal is to make such a software engineering model that makes the software with less effort and less resources with High quality. So, here we are going to generate new software development model that is based on waterfall and iterative approach.

Now we are going to present new model steps as below:

4.2 Proposed Steps:

1. Requirement Specification and analysis
2. Dividing into small modules
3. Discard module designing phase and combine it into coding phase
4. Assign that phase to the experienced developer.
5. Implement and test module (Advantage of Code and Fix model)
6. Deploy all alpha and beta version and collect review of it.
7. Deployment and maintenance
8. Ready for version up gradation.
9. Active interaction with customer for frequent requirement change

4.3 Exploring New SDLC Model

Here we have made new SDLC model for general use in industry named SDLC_2015 that is applicable for all kind of software requirements. Here we have compared our model with beginner's waterfall model and iterative development approach. We have taken a standard set of requirements and develop two applications with Waterfall, iterative and our new approach.
Let here discuss the major functions of new proposed model.

4.3.1 Communication & analysis

It is the same as all models where requirements are collected from end clients and analyzed as all feasibility studies.

4.3.2 Design & Coding

It is the major part of the proposed architecture where design part of SDLC is combined with coding department and coder is only responsible for the coding and designing as well and the major benefit is that concept distribution gap is not there.

4.3.3 Testing

Here white and black box testing is done.

4.3.4 Demo Release

The software is demo released for the use as office versions and beta review versions.

4.3.5 Release Versions

Developed software is released in alpha, beta and full version release.

4.3.6 Customer

Customer or client is in the center of all process where client is communicated at the development of all modules and revert back on requirement changes from clients.

4.3.7 Deployment

After success versions the software is deployed using the help of server engineer on the third party server or client own server.

4.3.8 Maintenance

Maintenance is the function where developed software is maintained by an administrator to take review and see new updates in software.

4.3.9 Gate Keeper

The role of gate keeper is to provide actual release of the software after public review and popularity measurement of alpha and beta versions and corrections.

4.3.10 Server Engineer

Server Engineer helps in the deployment process and software is deployed on the server with all configurations required.

4.3.11 Release Manager

Working of a release manager to maintain the software update versions and keep report of bug and errors to resolve in next version.

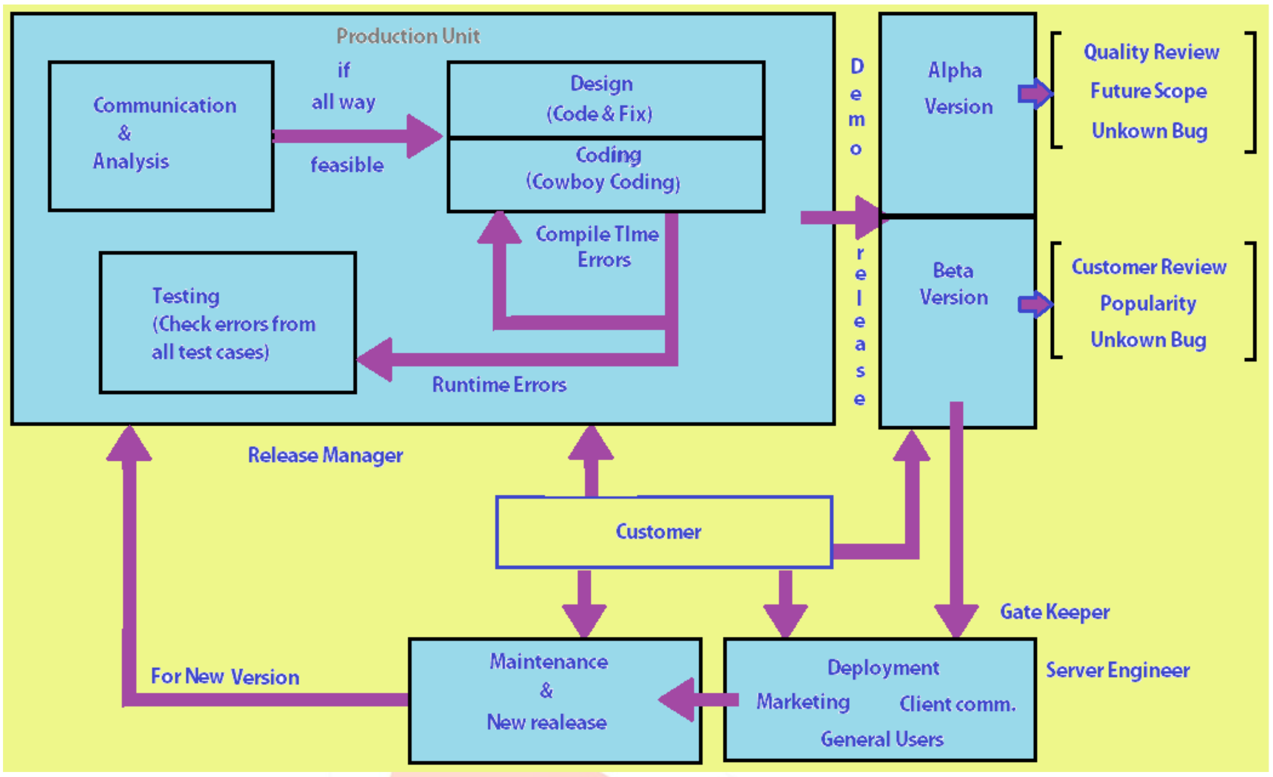


Fig-1 Proposed System Architecture

4.4 *Brief of the Developed Projects*

Two of the developed projects are OCAMS (Online Classified Ads Marking System (in which user can view ads and post their ads like OLX and quikr)) and OAPMS (Online Agriculture Products Management Systems (in which agriculture products are managed and make auction for the crop seeds and pesticides etc.).

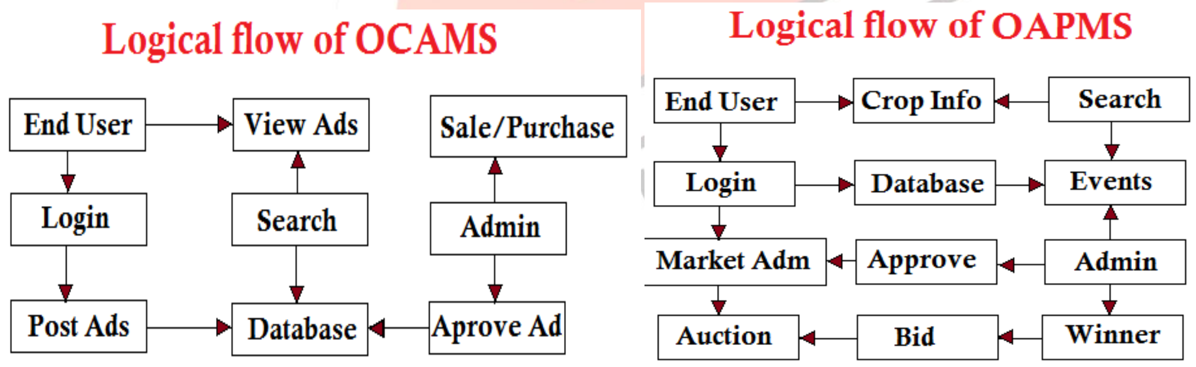


Fig-2 Logical Flows of the Projects

V. Exploring Results

Estimated	16000	14250	12250
Model	**Waterfall**	**Iterative**	**SDLC_2015**
KLOC	16	14.25	12.25
Effort	39.7	35.15	29.99
Time	10.13	9.67	9.1
Persons	4	4	3
Resource	14	13	12
Cost	44662.5	39543.75	33738.75
R_Cost	44660	39540	33740

Fig-3 COCOMO Estimation OCAMS

Estimated	15000	13500	9750
Model	**Waterfall**	**Iterative**	**SDLC_2015**
KLOC	15	13.5	9.75
Effort	37.1	33.21	23.6
Time	9.87	9.46	8.31
Persons	4	4	3
Resource	16	14	12
Cost	41737.5	37361.25	26550
R_Cost	41740	37360	26550

Fig-4 COCOMO Estimation OAPMS

OCAMS	Waterfall	Iterative	SDLC_2015
Resource	4	4	3
Man Power	4	4	3
Quality	361	322	293
Cost	43900	37900	31450
Profit	1100	7100	13550

Fig-5 Live project details OCAMS

OAPMS	Waterfall	Iterative	SDLC_2015
Resource	4	4	3
Man Power	4	4	3
Quality	340	304	281
Cost	41680	36090	26020
Profit	4250	8750	17050

Fig-6 Live project details OAPMS

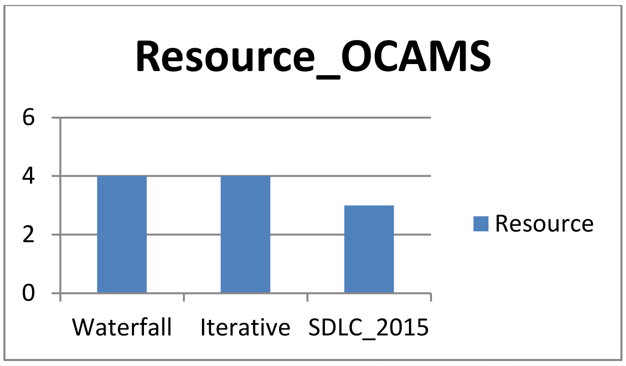
OCAMS		Waterfall			Iterative			SDLC_2015		
Module	**Ways**	**Total**	**Success**	**Ratio**	**Total**	**Success**	**Ratio**	**Total**	**Success**	**Ratio**
Login	10	15	8	53.3	15	11	73.3	15	14	93.3
Registration	7	15	9	60	15	12	80	15	14	93.3
Post	10	15	11	73.3	15	11	73.3	15	13	86.7
Approval	5	15	10	66.7	15	13	86.7	15	14	93.3
Category	10	15	12	80	15	13	86.7	15	14	93.3
Subcategory	10	15	12	80	15	12	80	15	13	86.7
Item Gallery	7	15	14	93.3	15	14	93.3	15	15	100
Filter & Sort	8	15	10	66.7	15	11	73.3	15	13	86.7
General	10	15	7	46.7	15	9	60	15	12	80

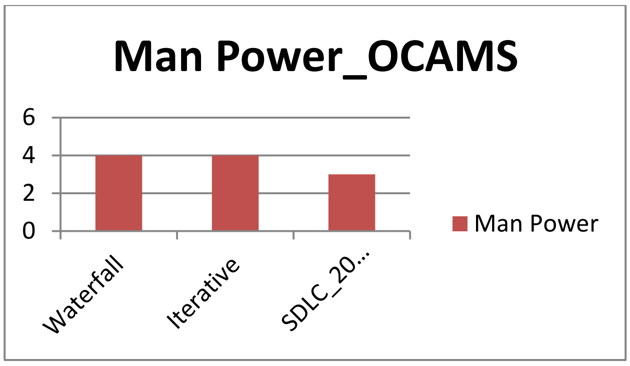
Fig-7 Various Test cases Results OCAMS

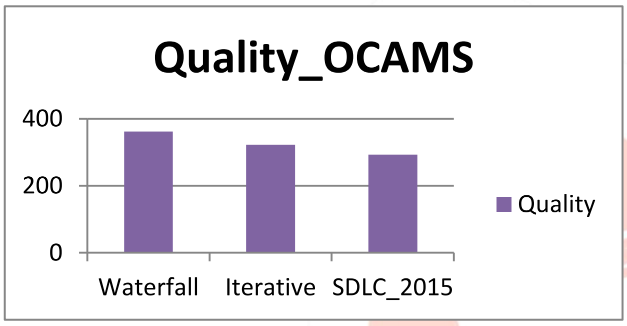
OAPMS		Waterfall			Iterative			SDLC_2015		
Module	**Ways**	**Total**	**Success**	**Ratio**	**Total**	**Success**	**Ratio**	**Total**	**Success**	**Ratio**
Login	10	15	9	60	15	11	73.3	15	14	93.3
Registration	7	15	10	66.7	15	12	80	15	14	93.3
Crop Information	5	15	11	73.3	15	12	80	15	15	100
Purchase Crop	10	15	10	66.7	15	11	73.3	15	13	86.7

Sell Crop	10	15	10	66.7	15	11	73.3	15	13	86.7
Auction Products	15	15	9	60	15	11	73.3	15	13	86.7
Bid	10	15	8	53.3	15	10	66.7	15	14	93.3
Categorical View	5	15	12	80	15	14	93.3	15	15	100
General	10	15	8	53.3	15	10	66.7	15	12	80

Fig-8 Various Test cases Results OAPMS







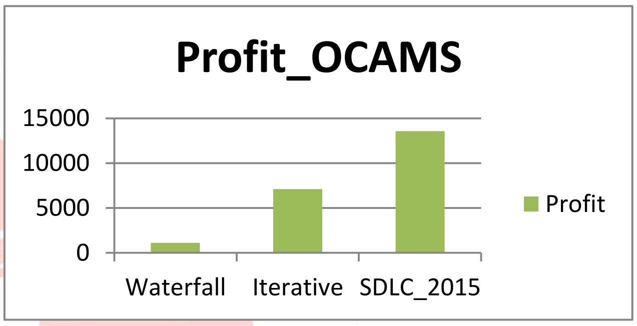
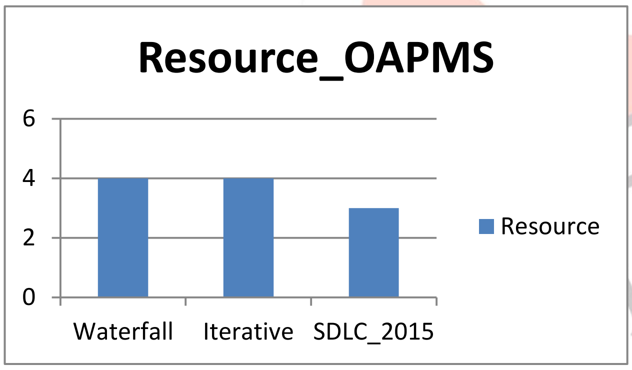
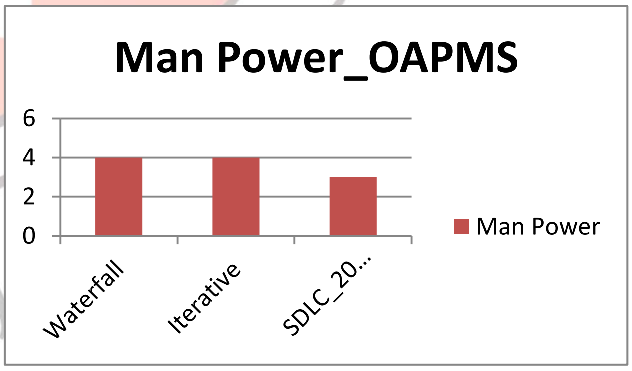
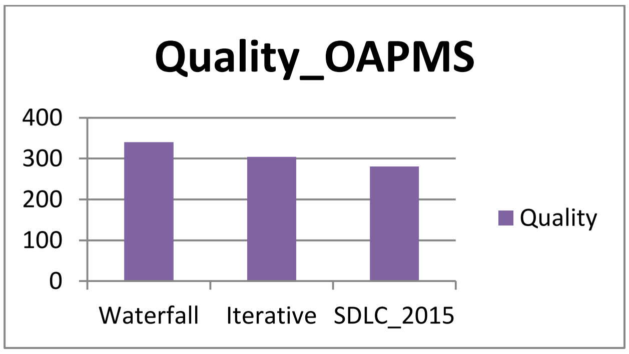


Fig- 9,10,11,12 Grpah analysis for OCAMS application







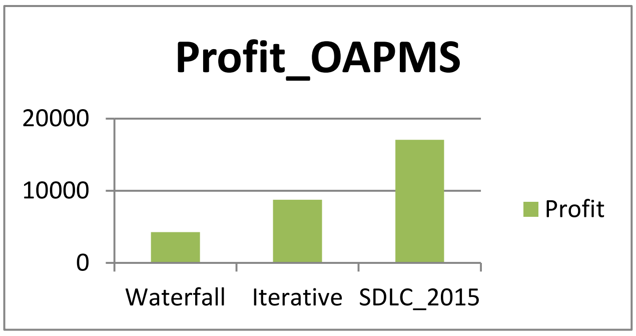


Fig- 13,14,15,16 Grpah analysis for OAPMS application

VI. Conclusion

At the end, after comparing and studying various approaches of software development, we came to know certain facts and parameter effects in software generation. The research goal is to make such a software engineering model that makes the software with less effort and less resources with same quality and less time. So, here we are going to combine designing and coding phase to generate new SDLC model and that can applicable for almost all kind of software requirements. The main advantage of combining designing and coding phase is to save time and resource and maintain quality of the software. By following the new model the software generation will be in less time with high quality and less effort and less resources. So this new approach will be useful the IT industry a lot.

REFERENCES

[1] Mr. Preyash Dholakia, Mr. Dishek Mankad, “The Comparative Research on Various Software Development Process Model”, International Journal of Scientific and Research Publications, Volume 3, Issue 3, March 2013, ISSN 2250-3153

[2] Ms. Shikha maheshwari, Prof.Dinesh Ch. Jain, “A Comparative Analysis of Different types of Models in Software Development Life Cycle”, IJARCSSE, Volume 2, Issue 5, May 2012

[3] Barry Boehm, Dan Port, Ye Yang,” Win-Win Spiral Approach to Developing COTS-Based Applications”, EDSER-5 Position Paper, University of Southern California, Texas A&M University

[4] Walt Scacchi, ”Process Models in Software Engineering”, Institute for Software Research, University of California, Irvine, February 2001, Revised Version, May 2001, October 2001, Final Version to appear in, J.J. Marciniak (ed.), Encyclopedia of Software Engineering, 2nd Edition, John Wiley and Sons, Inc, New York, December 2001.

[5] Nabil Mohammed Ali Munassar and A. Govardhan, “A Comparison between Five Models Of Software Engineering”, IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 5, September 2010, ISSN (Online): 1694-0814

[6] Barry Boehm, edited by Wilfred J. Hansen, “Spiral Development: Experience, Principles and Refinements, Spiral Development Workshop”, February 9, 2000, July 2000

[7] C. Abts, B. Boehm, and E. Bailey Clark, “COCOTS: A Software COTS-Based System (CBS) Cost Model” Proceedings, ESCOM 2001, April 2001, pp. 1-8.

[8] C. Albert and L. Brownsword, “Evolutionary Process for Integrating COTS-Based Systems (EPIC): An Overview” CMU-SEI-2002-TR-009, July 2002.

[9] R. Balzer, “Living with COTS” Proceedings, ICSE 24, May 2002, p. 5.

[10] Gupta, A., I.S. Mumick, “Maintenance of Materialized Views: Problems, Techniques, and Applications.” Data Eng. Bulletin, Vol. 18, No. 2, June 1995. 9 Zhuge, Y., H. Garcia-Molina, J. Hammer, J. Widom, “View Maintenance in a Warehousing Environment, Proc. Of SIGMOD Conf., 1995.

[11] Roussopoulos, N., et al., “The Maryland ADMS Project: Views R Us.” Data Eng. Bulletin, Vol. 18, No.2, June 1995. [11] O’Neil P., Quass D. “Improved Query Performance with Variant Indices”, to appear in Proc. of SIGMOD Conf., 1997.

[12] Ian Sommerville, "Software Engineering", Addison Wesley, 7th edition, 2004.

[13] CTG. MFA – 003, "A Survey of System Development Process Models", Models for Action Project: Developing Practical Approaches to Electronic Records Management and Preservation, Center for Technology in Government University at Albany / Suny, 1998.

[14] Steve Easterbrook, "Software Lifecycles", University of Toronto Department of Computer Science, 2001.

[15] National Instruments Corporation, "Lifecycle Models", 2006, http://zone.ni.com.