An Effective Evaluation of Software Development Models Amalgamation

V. Therese Clara
Asst professor of Computer Science, Madurai Kamaraj University College, Madurai, India

ABSTRACT

In the ever changing software development environment, the software developer faces many challenges. It is highly advantageous to synergize various models in order to optimize software development. The feasibility of combining project characteristics of different models will enable better outcomes. To measure project characteristics of compatible models, values are assigned to characteristics. The values are represented in bar diagrams. The interpretation of suggested combination of models made through bar diagrams is proved using Operations Research.

Keywords

Synergize, optimize, feasibility, characteristics, compatible, interpretation.

1. INTRODUCTION

Software process is a framework to build high quality software [1]. Software Process Model becomes an abstract representation of a software process. Each process model represents a process from a particular perspective, and thus provides only partial information about that process [2]. Selecting an appropriate software process model, completely suitable to a particular situation is a difficult task in software engineering. Improper selection of software process model results in creating a bottleneck for the software product, consuming more time and overshooting the budget. Hence care must be taken during a selection of software process model. The advantage of selecting the right software process model enhances the ability and quality product. The realization is found within a stipulated budget and time. Nowadays it is quite impractical to follow exclusively one model for some difficult situations. In this regard project characteristics of different models are appropriately synthesized in creation of a hybrid model. This paper proposes an approach to select an appropriate combination of SDLC models based on different project characteristic categories.

2. NEED FOR INTEGRATION

The competitive development scenario and the survival drive necessitate the possibility of having more than one approach in various phases of development in the Software project development methodology. Following a single model for the development in the present software industry is at times not feasible. Development firms have begun to explore the possibilities of combining the development models in order to produce software to meet global software demands. In the event of integration of software development models there could be clashes or repetition of the processes. This research paper proposes a method to integrate software development models.

3. PROPOSED WORK

3.1 Algorithm

1. Be familiar with various models
2. Review and analyze the types of work performed
3. Review the life cycle approach of each model
4. Identify the set of features and attributes
5. Evaluate the effectiveness of the life cycle frame work
6. List the features of various models in Software Development models
7. Assume the rating for their characteristics for each model
8. Sum up the rating score of each model
9. Integrate this rating to form another table
10. Apply Operations Research (Hungarian algorithm)
    a) Row reduction
    b) Column reduction
    c) Strike out the Maximum number of zeros in the rows and columns of table
    d) Assign the value of combined model
    e) Put the rank
    f) Suggest the best combination of models
11. Produce best suggested combination of software model after the process of Assignment model

3.2 Concept Application:

There are several different approaches to software development, like various views regarding governance of a country. A more structured engineering-based approach is suitable to developing business solutions, whereas a more incremental approach is recommended in a situation where software is developed piece-by-piece. Most methodologies share some combinations of the following stages of software development: market research, analyzing the problem, implementation of the software, testing the software, deployment, maintenance and bug fixing [3]. Table-1 shows the comparison of different models on the basis of certain features /factors which may influence the selection of lifecycle models [4].

Table – 1

Model / Features	Waterfall Model	Prototype Model	Spiral Model	Iterative Model	Agile Model	RAD
Requirement Specifications	1	3	1	1	3	3
Understanding Requirements	5	2	5	2	5	6
Cost	1	5	3	1	6	1
Guarantee of Success	1	3	5	5	6	3
Resource Control	6	0	6	6	0	6
Cost Control	6	0	6	0	6	6
Simplicity	6	6	3	3	1	6
Risk Involvement	0	1	6	1	1	6
Expertise Required	5	3	5	5	6	3
Changes Incorporated	1	6	6	6	1	6
Risk Analysis	1	6	5	6	5	6
User Involvement	1	5	5	3	5	1
Overlapping Phases	0	6	6	0	6	1
Flexibility	3	6	5	0	6	5
Total	37	52	67	39	57	59

Requirement specifications are needed just at the beginning of the Waterfall model, Spiral model and Iterative model. Waterfall model, spiral model and agile models need good understanding of the requirements, while prototype model and Iterative model do not need good understanding of the requirements [10]. Waterfall and Iterative models are used for projects, which have low cost requirements, while the agile process model leads to very high cost [11]. There is not much resource control in Prototype and Agile models. On the other hand, there is resource control in Waterfall, Spiral and Iterative models. The guarantee of success, is low in Waterfall model, relatively good in Prototype model. The intermediate guarantee of success strikes a mid path between good and high. However the degree of success is very high while using agile model. Data values for agile, Waterfall and Spiral have cost control feature as a control factor which is quite significant for software projects. The Spiral and Iterative models have limited impact because they are intermediate with regard to simplicity factor, while the Agile model is unsuitable because of complex nature. Also because of its complexity, more time and money is required to complete a software project [12]. As Spiral model has low risk and Waterfall model has high risk, Spiral model is preferred. Prototyping models are highly suitable for developers while Waterfall, Spiral, and Iterative are unsuitable because they demand high expertise and experience. After the completion of the projects less changes are made while using Prototype, Spiral and Iterative models. While the Waterfall model and Agile model are totally inappropriate because if they require the changes to be incorporated, then many difficulties do arise while incorporating changes in the software project [13].

The model features are rated between 0 and 6. The total for each model is calculated by adding the values attributed to the characteristics. These ratings are represented using bar charts for each model . Fig .1 shows the bar chart for the waterfall model which incorporates the characteristics.

3.2.1 Bar chart Analysis

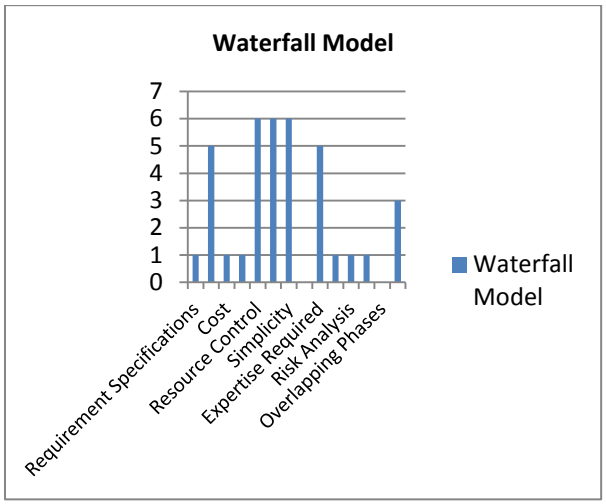

Fig 1

The bar chart is represented for each charactreistic for all the six models. A sample is shown for Requirement Specifications as in Fig-2 and User Involvement as in Fig - 3.

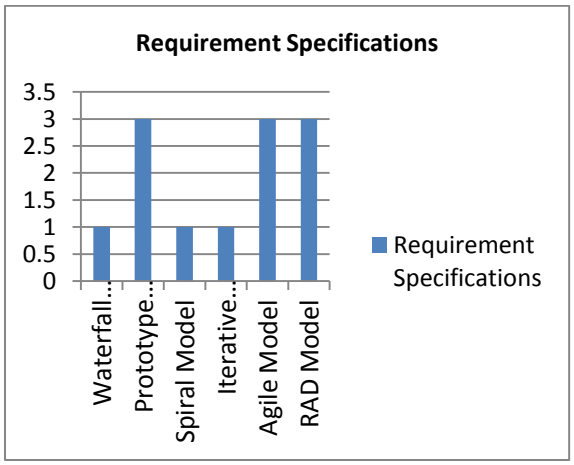

Fig -2

From the above analysis the Requirement Specifications are required at the beginning for Waterfall, Spiral and Iterative models whereas for Agile, Prototype and RAD models the requirement specifications are frequently changed during the software development [9]. It is inferred that it is possible to

combine Prototype or Iterative or Agile models when the Requirement specifications are changing.

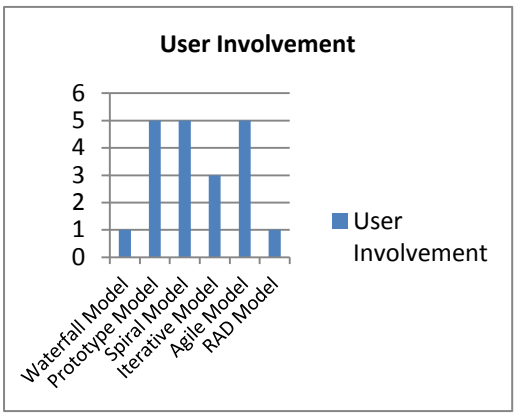

Fig.3

Considering the User Involvement for Waterfall and RAD model, it is very less throughout the development. Iterative model needs intermediate user involvement, whereas Spiral model and Agile process models require high user involvement as a requirement of these models [8]. So it is possible to combine Prototype or Spiral or Agile because the User Involvement is high in all the three cases.

3.2.2. Result of Bar Chart Analysis

Table - 2 depicts the result of bar chart analysis. The suggested combination of models based on the features of the project is given below.

Table - 2

Project Features	**Suggested Combination of Model**
Requirement Specifications	Prototype & Agile
Understanding Requirements	Water ,Spiral ,Agile, RAD
Cost	Agile
Guarantee of Success	Spiral, Iterative, Agile
Resource Control	Water ,Spiral ,Iterative, RAD
Cost Control	Water, Spiral, Agile, RAD
Simplicity	Water, Prototype, RAD
Risk Involvement	Spiral, RAD
Expertise Required	Agile, Water, Spiral, Iterative
Changes Incorporated	Prototype , Spiral, Iterative, RAD
Risk Analysis	Prototype, Iterative, RAD
User Involvement	Prototype, Spiral, Agile
Overlapping Phases	Prototype, Spiral , Agile
Flexibility	Prototype, Agile

3.2.3 Applying Hungarian Algorithm

The above analysis is proved using Assignment Problem. The total value of the characteristics of each model is calculated. The objective is to combine any of Waterfall, Prototype, Spiral, RAD, Agile and Iterative models. The total value of each model is assigned in both the column and row of the data matrix. The most beneficial combination of models is assessed. The Hungarian method by Mr. Koning of Hungary or the Reduced matrix method is used for solving Assignment Problems [6]. The Hungarian method is an algorithm which finds an optimal assignment for a given matrix. Row reduction and column reduction operation has to be performed until all zeros have either been assigned or crossed-out.

The reduced matrix will then have nonnegative elements with at least one zero in each row and in each column. If it is possible to find a set of n independent zeros, then an assignment among the independent zeros will provide an optimal solution to the problem [7]. One of the significant challenges while applying Assignment Problem is the optimality of solution. The certainty of the best available solution is skeptical. Whereas in the Hungarian Algorithm, the test for optimality is quite straightforward and follows drawing of lines through the zeros in the Opportunity Cost Matrix. When the number of lines (horizontal and vertical only) equals the number of rows/columns, the solution is optimal. On the other hand, when the number of lines is not equal to the number of rows/columns, the solution is not optimal but still it can be improved. At times it may be considered to be a confusing method superficially, but its straightforward nature facilitates an easy implementation and clarity of interpretation. The researcher has used this method to explore the possible optimal solution for combining models.

4. THE RESULT OF HUNGARIAN ALGORITHM

While considering the model features it is possible to combine select models. For instance, while selecting the project feature of understanding requirements, chances are there for a hybrid Spiral-Agile model. Depending on another project feature like simplicity it is quite feasible to synthesize models like Waterfall and Prototype as one combination and Prototype and RAD as another hybrid variety. Similarly other project features lend themselves for an easy merging of two or more models. Such possible combination of appropriate models is listed in the following recommendation as shown in figure 4.

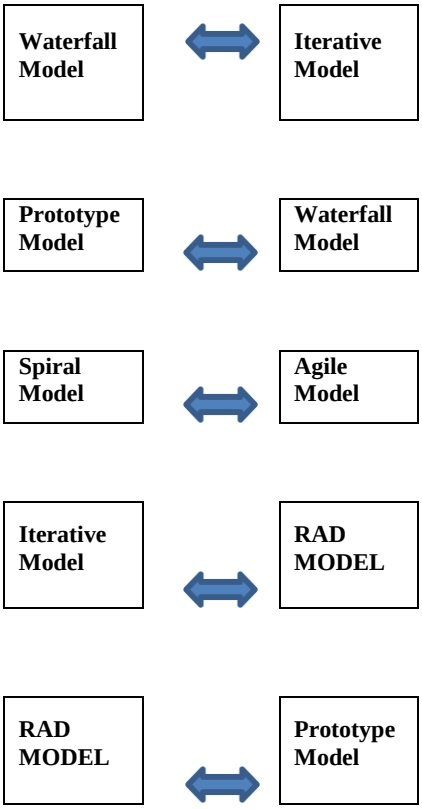

Fig. 4

From the result of Hungarian Algorithm various combinations of models are suggested when the characteristics of the project are considered. Thus, the results of Bar chart analysis are proved using the Assignment problem.

5. CONCLUSION

In the highly volatile software development scenario, it has become imperative to not only follow one particular model but also have hybrid models. Some characteristics of a model have an edge over others. Such prominent project characteristics in each model are chosen and their compatibility is checked using the assigned values to validate the rationale for selection. The project characteristics of some models facilitate an easy integration for boosting performance. The selection of possible combination of models based on project characteristics is made feasible with the bar chart analysis. It is proved using Operations Research.

6. REFERENCES

[1] Pressman R. S., *Software Engineering a Practitioner's* Approach, Fifth Edition, McGraw Hill, 2001.

[2] Ian Sommerville, "Software Engineering", 8th Edition, 2006, pp. 89.

[3] Kushwaha ety.al, "Software Development Process and Associated Metrics - A Framework", IEEE CNF, Page(s): 255 – 260, July 2006.

[4] Dr. Deepshikha Jamwal, "Analysis of Software Development Models" IJCST Vol. 1, Iss ue 2, December 2010

[5] Manish Sharma, "A Survey of project scenario impact in SDLC models selection process", International Journal of Scientific & Engineering Research Volume 2, Issue 7, July-2011

[6] Prem Kumar Gupta, Dr, D.S. Hira , "Operations Research", pp 326.

[7] Sinha, S.M "Mathematical Programming Theory and Methods" Pages 278

[8] Chan, D.K.C. Leung, K.R.P.H, "Software development as a workflow process", 2-5 Page(s): 282 – 291, Dec. 1997.

[9] Molokken-Ostvold et.al, "A comparison of software project overruns - flexible versus sequential development models", Volume 31, Issue 9, Page(s): 754 – 766, IEEE CNF, Sept. 2005.

[10] Boehm, B. W. "A spiral model of software development and enhancement", ISSN: 0018-9162, Volume: 21, Issue: 5, on page(s): 61-72, May 1988.

[11] Abrahamsson P. et.al, "Agile Software Development Methods: Review and Analysis", ESPOO, VTT Publications 478, VTT Technical Research Centre of Finland. http:/www. fi/pdf/publications/2002/P478.pdf, 2002.

[12] Abrahamsson, P. et.al, "New Directions on Agile Methods: a Comparative Analysis", The Proceedings of the 25Th International Conference on Software Engineering. Pp. 244-254, 2003.

[13] Davis, A.M et.al, "A strategy for comparing alternative software development lifecycle models", Software Engineering, IEEE Transactions on Volume 14, Issue 10, Oct 1988.