Ontology Development and Analysis for Software Development Life Cycle Models

Sandeep Kumar, Surinder Singh, Kuldeep Kumar, Ankita Jain, and R. B. Mishra

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 such as waterfall model, iterative waterfall model, prototyping model, spiral model etc. These models have their own unique characteristics and are suited to a particular situation of software development and software types. One software life cycle model may prove to be more efficient than the other one depending upon the development environment. In this paper, the work has been done to analyze the various software life cycle models from this aspect. Further, the work has been done to develop the ontology for various categories of the software projects and an observation on software life cycle models according to these categories. The ontology has been developed in the Semantic Web ontology development language, Web Ontology Language (OWL).

Keywords- Categories, Ontology, Taxonomy, Software project, SDLC.

I. INTRODUCTION

A *software project*, regardless of whether it is large or small, goes through certain defined stages, which together, are known as the Software Development Life Cycle (SDLC) [4]. There are five phases that are the part of the SDLC. These phases are: requirements definition, design, coding, testing, and maintenance.

SDLC models are created based on the various phases of the SDLC, the order in which they occur and the interaction between them. The output generated by each phase serves as the input for the next [14]. Some of the SDLC models are discussed in the next section. One software life cycle model may prove to be more efficient than the other one depending upon the development environment. The paper analysis the various SDLC models from this point of view. The paper also presents the *ontological* classification of various software projects.

Apart from Introduction in Section-1, the work has been organized as follows. Section-2 presents the brief of various SDLC models. The suitability analysis of various models has been done in Section-3. In section-4, a broad classification of various types of software projects has been presented. The section also presents its *ontological* presentation. Section-5 presents our observations from the work and the work has been concluded in the Section-6.

Sandeep Kumar is with the Department of Electronics and Computer Engineering, Indian Institute of Technology, Roorkee-247667, India (sandeepkumargarg@gmail.com, sgargfec@iitr.ernet.in).

Surinder Singh, Department of Information Technology, M. M. Engineering College, M. M. University, Mullana, India

Kuldeep Kumar, Department of Computer Engineering, National Institute of Technology, Kurukshetra, India.

Ankita Jain, Department of Electronics and Communication Engineering, National Institute of Technology, Kurukshetra, India

R. B. Mishra, Department of Computer Engineering, Institute of Technology, Banaras Hindu University, Varanasi, India

II. SDLC MODELS

A. Waterfall Model

Waterfall model is a classical approach, widely used in Software Engineering to ensure success of the project [4, 14].

B. Prototyping Model

Prototyping Model quickly develops a working model that is functionally equivalent to a component of the project [4, 14].

C. RAD (Rapid Application Development)

RAD is a concept that products can be developed faster and of higher quality by using some special techniques [4, 14].

D. Incremental Model

It constructs a partial implementation of a total system. Then slowly add increased functionality [4, 14].

E. Spiral Model

This model combines the features of the prototyping model and the waterfall model; it adds risk analysis, and 4gl RAD prototyping to the waterfall model [4, 14].

F. XP (Extreme Programming) Model

XP model is used for small-to-medium-sized teams developing software with vague or rapidly changing requirements. Coding is the key activity throughout a *software project* [1].

G. Scrum Model

It is similar to other life cycle models that use iterative development to address changing requirements, but in it, the repetitions are referred to as sprints, which normally last thirty days [1].

Different SDLC models have their unique characteristics and requirements. On the basis of these aspects, this paper presents an analysis survey on the suitability of various SDLC models on the situation of its use for *software project* development. Further, an *ontology* is developed for the various categories of the *software projects* and an observation is presented on software life cycle models according to these categories. The *ontology* has been developed in the Semantic Web ontology development language, Web Ontology Language (OWL).

III. Suitability Analysis Of Various SDLC Models

The Table I discusses the suitability analysis of various SDLC models on the situation of its use for software project development.

TABLE I
SUITABILITY ANALYSIS OF VARIOUS SDLC MODELS

SDLC	Type of Situation in which it is Suitable
Waterfall	-Fixed requirements. -Work proceeds to completion in a linear manner. -Without any time aggressiveness. -It can not handle High Risks Project [6, 12].
XP	-Small projects. -Full time user representative. -It will not work if the team must predict the cost and schedule [6, 12].
Scrum	-Large development projects. -Projects using object-oriented technology [6, 12].
RAD	-Time line is aggressive. -Risk is not so high. -Small to medium size projects. [6, 12].
Prototyping	-Developers get to build something immediately. -It cannot handle high level of risks [6, 12].
Incremental	-High technical risks. -Time line is aggressive [6, 12].
Spiral	-Large –scale system and software. -Level of Reliability is High. --Understand and react to Risks at each evolutionary level [6, 12].

IV. Broad Classification Of Software Projects And Its Ontology

A. Classification of Software Projects

Software projects are classified into different categories on the basis of their features like use of the software project, no. of line of codes, no. of components used, geographical condition, hardware platform, user requirements etc. [9,10,11].

The t*axonomy* of software projects are presented in Fig. 1. According to *taxonomy* the description of different categories are given below.

1. First Level Categories

i. *Application Software Project* consists of standalone programs that solve a specific business need [8].

ii. *System Software Project* is a collection of programs written to service other programs [8].

iii. *Embedded Software Project* resides within a product or system and is used to implement and control features and functions for the end-user and for the system itself [8].

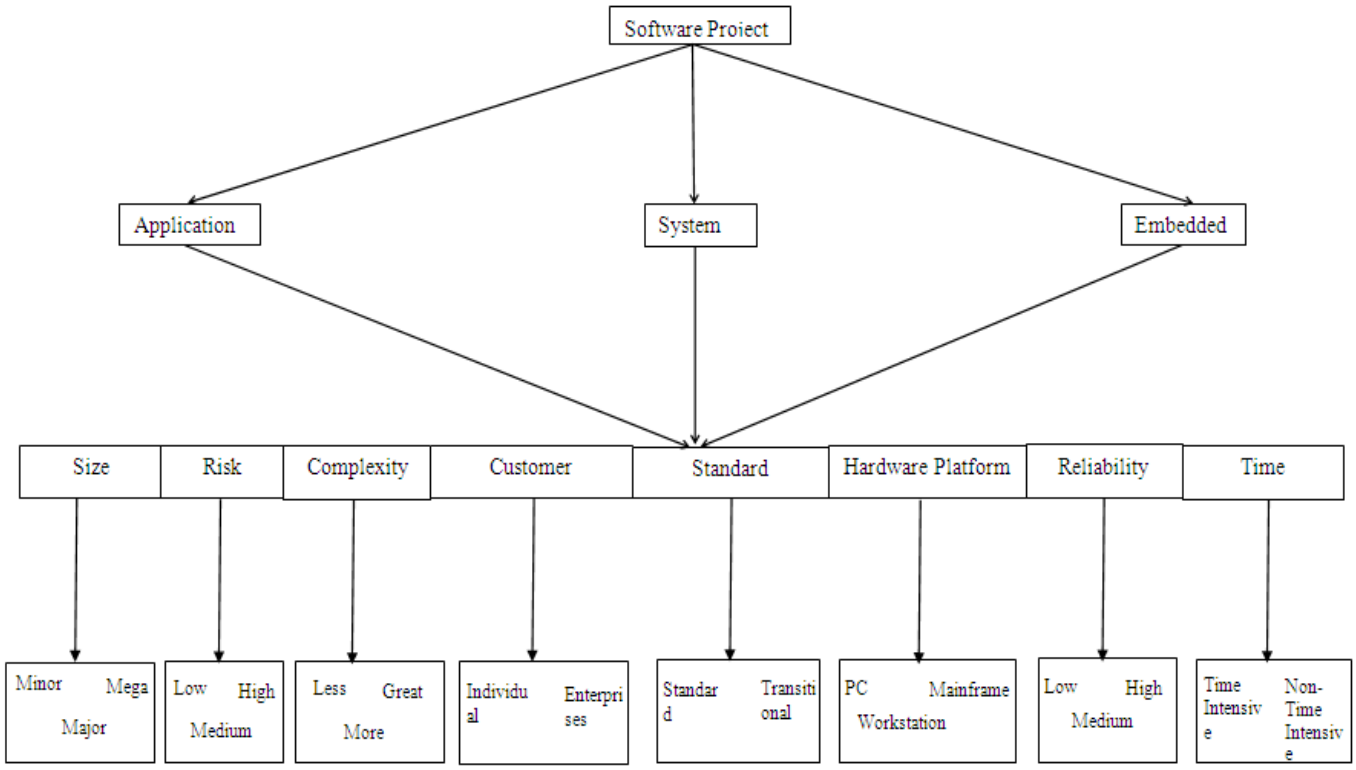

Fig. 1 Taxonomy of Software Project

2. Second and Third Level Categories

i. *Size* is defined as the level of effort associated with development and/or maintenance of the software [3]. This category is further divided into three sub categories:- Minor, Major and Mega.

ii. *Risk* is any potential situation or event that could negatively affect a project's ability [13]. This category is further divided into three sub categories:- Low, Medium and High.

iii. *Complexity* of software project depends on the resources expended by a system while interacting with a piece of software to perform a given task [7]. Complexity is sub divided into three categories:- Less, More and Great.

iv. *Customer* means who is the user of the software project, is it an Individual person or an Enterprise [10].

v. *Standard,* It may be useful to look at projects from the perspective of how "Standard" they are versus how much change they bring to their owners, their sponsoring company or agency, or the affected economy as "Transitional" projects.

vi. *Hardware Platform* is an important issue that where the software project will work. Is this designed for a single PC or for a Workstation or for a Mainframe [8]?

vii. *Reliability Level* software reliability is defined as: the probability of failure-free software operation for a specified period of time in a specified environment [11]. Reliability has three levels: - Low, Medium and High.

viii. *Time* depends on estimated time line for the software project [10]. Project manager must know about the time line that is the time line is 'Intensive' or' Non-Intensive'.

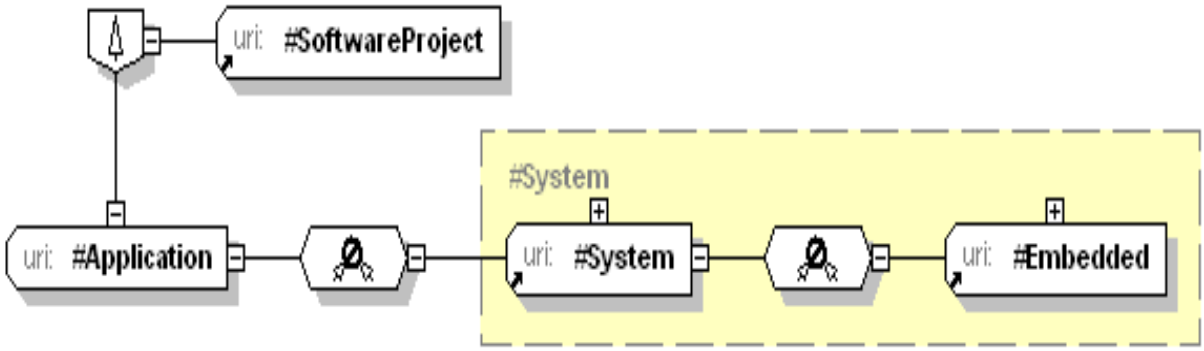

Fig. 2 Ontology of Taxonomy's First Level

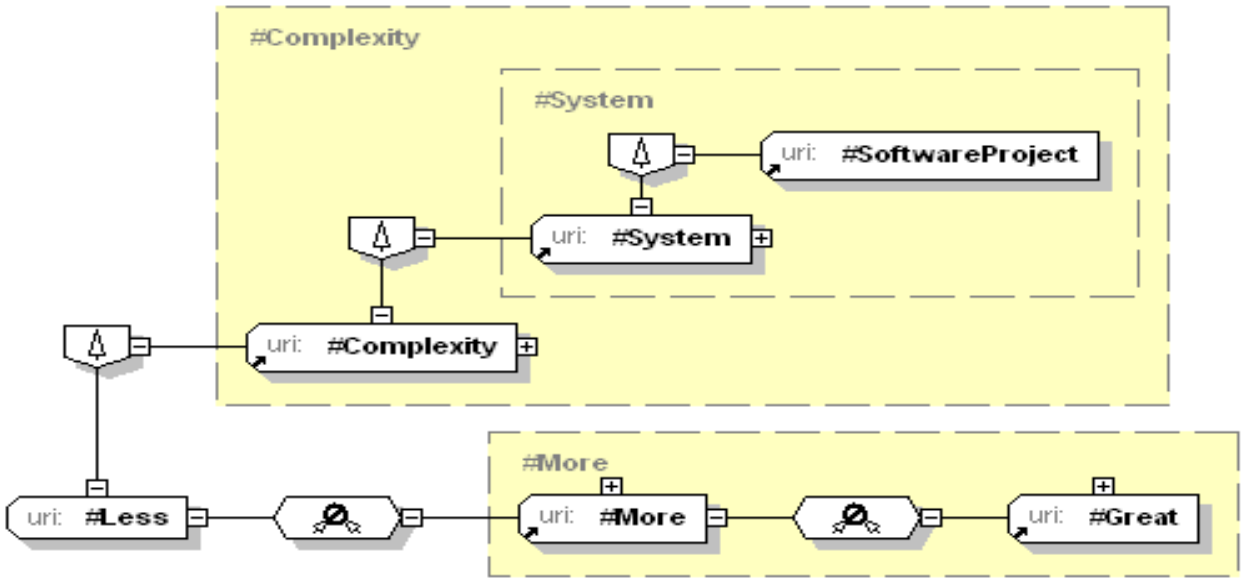

Fig. 3 Ontology of Taxonomy's 2nd and 3rd Level

B. Ontological Representation of Proposed Taxonomy

Ontological representation means a graphically hierarchical representation of any taxonomy.

On the basis of above taxonomy of the software categories, this paper presents ontology by using *JENA* library in *Java*. The ontology designed in *OWL* language and Then graphically represented using *Altova* software. The designed ontology is well formed and full.

Fig. 2 shows an ontological representation of the first level of the presented Taxonomy. Where the software project have

Three main categories of the *software projects* are Application, System and Embedded. As the figure shows that 'software project' is the root node and the 'Application', 'System' and 'Embedded' are the child node of the root node.

Fig. 3 shows an ontological representation of the second level and one category of the third level of the presented Taxonomy, where the 'software project' main categories 'System' and its subcategory 'Complexity', then its subcategories 'Less', 'More' and 'Great'.

V. OBSERVATION

After analyzing the SDLC models and classifying the software, we found that there is a suitable model for different type of software. Observation table of SDLC models according presented taxonomy is given in Table II.

TABLE II
OBSERVATION TABLE OF SDLC MODELS

S.No.	Situations	SDLC Models
1.	Where the Type of the software is Application and Size is Minor and Risk is low and Complexity is Less and Customer is Individual and Standard is Standard and Hardware Platform is PC and Reliability is Low and Time is Non-Intensive.	Waterfall/XP.
2.	Where the Type of the software is Application and Size is Mega and Risk is low and Complexity is Less and Customer is Individual and Standard is Standard and Hardware Platform is PC and Reliability is Low and Time is Non-Intensive .	Waterfall/Scrum
3.	Where the Type of the software is System and Size is Minor and Risk is low and Complexity is Less and Customer is Individual and Standard is Standard and Hardware Platform is PC and Reliability is High and Time is Intensive.	RAD/Prototyping
4.	Where the Type of the software is System and Size is Minor and Risk is High and Complexity is Great and Customer is Individual and Standard is Standard and Hardware Platform is PC and Reliability is High and Time is Intensive.	Incremental/Spiral
5.	Where the Type of the software is Embedded and Size is Mega and Risk is Low and Complexity is More and Customer is Individual and Standard is Standard and Hardware Platform is PC and Reliability is High and Time is Intensive.	Scrum/ RAD
6.	Where the Type of the software is Embedded and Size is Mega and Risk is High and Complexity is Less and Customer is Individual and Standard is Standard and Hardware Platform is Mainframe and Reliability is High and Time is Non-Intensive.	Spiral/Incremental
7.	Where the Type of the software is Application and Size is Minor and Risk is High and Complexity is Less and Customer is Individual and Standard is Standard and Hardware Platform is Workstation and Reliability is Medium and Time is Intensive.	Incremental/Spiral
8.	Where the Type of the software is Application and Size is Mega and Risk is High and Complexity is Less and Customer is Enterprises and Standard is Standard and Hardware Platform is PC and Reliability is High and Time is Intensive.	Spiral/Incremental
9.	Where the Type of the software is Application and Size is Mega and Risk is low and Complexity is Great and Customer is Enterprises and Standard is Standard and Hardware Platform is PC and Reliability is Low and Time is Intensive.	Scrum/ RAD
10.	Where the Type of the software is Application and Size is Minor and Risk is Medium and Complexity is More and Customer is Individual and Standard is Standard and Hardware Platform is Workstation and Reliability is Medium and Time is Intensive	RAD/Prototyping

VI. CONCLUSION

In this paper, the work has been done to analyze the various SDLC models with respect to their suitability for development of various types of software projects depending upon the development environment. The work has also been done to generate a broad classification and further its ontological representation of different types of software projects. Some observations have been taken to conclude the type SDLC models suitable for various development environments different project categories.

REFERENCES

[1] Abrahamsson, Pekka, Salo, Outi, Ronkainen, Jussi & Warsta, Juhani, "Agil software development methods. Review and analysis", Espoo, VTT Publications, 2002, pp. 107-110.

[2] Executive Brief, "Which Life Cycle Is Best For Your Project?", www.executivebrief.com/project.../best-project-life-cycle/ - United States, accessed on November 14, 2010.

[3] H. Gumuşkaya, "Core Issues Affecting Software Architecture in Enterprise Projects", World Academy of Science, Engineering and Technology 9- 2005. pp 32-37.

[4] I. Sommerville, "Software Engineering, Software Engineering", 7th edition, Addison-Wesley,2004, Reading, MA.

[5] J. Rothman, "What Lifecycle? Selecting the Right Model for Your Project", *Cutter IT Journal*, Vol 21, #5, May 2008.
[6] K. Schwalbe, "Information Technology Project Management", 6th edition, Cengage Learning, 2009.
[7] M. A. A. Ahmar, "Rule Based Expert System For Selecting Software Development Methodology", *Journal of Theoretical and Applied Information Technology*, 2005, pp 143-148.
[8] M. Cusumano, A. MacCormack, C. F. Kemerer, W. Crandall, "A Global Survey of Software Development Practices", MIT, Harvard University, University of Pittsburgh, Hewlett-Packard, Version 3.1 June 17, 2003, Forthcoming, IEEE Software.
[9] PMP, Fellow PMI, R. D. Archibald, and APM/IPMA, "Life Cycle Models For High-Technology Projects– Applying Systems Thinking To Managing Projects", Presented to the PMI-Central Iowa Chapter, Professional Development Day, Polk County Convention Complex, Des Moines, Iowa, October 17, 2003.
[10] R. D. Archibald and V. I. Voropaev, "Commonalities and Differences in Project Management Around The World: A Survey Of Project Categories And Life Cycles", *17th IPMA World Congress*, Moscow, June 4-6 2003.
[11] R. D. Banker, S. M. Datar, and C. F. Kemerer, "Factors Affecting Software Maintenance Productivity: An Exploratory Study", Carnegie Mellon University and Sloan School of Management Massachusetts Institute of Technology. Management Science, Vol 37, No. 1, printed in USA, January 1991, Pp 160-175.
[12] R. S. Pressman, "Software Engineering, A Practitioner's Approach", Sixth Edition, Mc Graw. Hill, 2005.
[13] R Lagerstr, L. M. V. W¨urtemberg, H. Holm, and O. Luczak, "Identifying Factors Affecting Software Development Cost", Industrial Information and Control Systems, the Royal Institute of Technology Stockholm, Sweden, Apr. 2010.
[14] S. Bhattacharjee, "Software Development Software Development Life Cycle", MOF, College of Agricultural Banking, RBI, PUNE, ab.org.in/Lists/ Knowledge%20Bank/ Attachments/ 83/, accessed on November 12, 2010.