RAMAIAH UNIVERSITY OF APPLIED SCIENCES

ASSIGNMENT

Course Code	19CSC212A
Course Name	Software Development Fundamentals
Programme	B. Tech
Department	Computer Science & Engineering
Faculty	Faculty of Engineering Technology

Name of the Student	Tanishq R Porwar
Reg. No	18ETCS002131
Semester/Year	4th / 2020
Course Leader/s	Ms Sahana P. Shankar

Declaration Sheet			
Student Name	Tanishq R Porwar		
Reg. No	18ETCS002131		
Programme	B. Tech	Semester/Year	4th / 2020
Course Code	19CSC212A		
Course Title	Software Development Fundamentals		
Course Date		To	
Course Leader	Ms. Sahana P. Shankar		

Declaration

The assignment submitted herewith is a result of my own investigations and that I have conformed to the guidelines against plagiarism as laid out in the Student Handbook.All sections of the text and results, which have been obtained from other sources, are fully referenced.I understand that cheating and plagiarism constitute a breach of University regulations and will be dealt with accordingly.

Signature of the Student		Date	
Submission date stamp (by Examination & Assessment Section)			

Signature of the Course Leader and date	Signature of the Reviewer and date

Contents

Engineering and Technology			
Ramaiah University of Applied Sciences			
Department	Computer Science and Engineering	Programme	B. Tech.
Semester/Batch	4th/2019		
Course Code	19CSC212A	Course Title	Software Development Fundamentals
Course Leader(s)	Ms. Sahana P. Shankar, Ms. Supriya and Ms. Prakash P		

Questions	Marking Scheme		Marks		
			Max Marks	First Examiner Marks	Moderator
1	1.1	Problems with the existing software process models for the current and future software development requirements	3		
	1.2	Design of a software process model which suits the current and future software development requirements	5		
	1.3	Justification of the designed model with an Example	2		
		Question 1Max Marks	**10**		
		Total Assignment Marks	**10**		

Course Marks Tabulation

Question	First Examiner	Remarks	Moderator	Remarks
1				
Marks (Max 10)				

Signature of First Examiner Signature of Moderator

Question No. 1

Solution to Question No. 1:

A1.1 Problems with the existing software process models for the current and future software development requirements

Software Development Life Cycle (SDLC) is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software.
The software company or the team of software developers have choice to select the SDLC model. Each of these models has its own weaknesses and strengths in different situations and circumstances. The challenge is to select which model should be good under certain conditions.

The following are the various stages of a typical SDL:

- Planning and Requirement Analysis
- Defining Requirements
- Designing the Product Architecture
- Building or Developing the Product
- Testing
- Deployment
- Maintenance

There are various software development life cycle models defined and designed which are followed during software development process. Each process model follows a series of steps unique to its type, in order to ensure success in process of software development. Following are the most important and popular SDLC models followed in the industry:

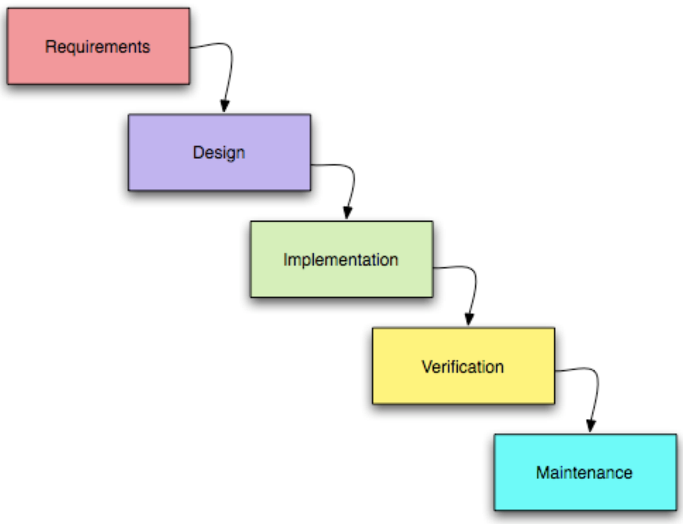
- Waterfall Model
- RAD Model
- Prototype Model

Waterfall Model

The Waterfall Model was first process model to be introduced provided by Winston W. Royce in 1970. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. The waterfall model illustrates the software development process in a linear sequential flow; hence it is also referred to as a linear-sequential life cycle model. This means that any phase in the development process begins only if the previous phase
is complete. In waterfall model phases do not overlap. In the waterfall approach, the whole process of software development is divided into separate phases. In waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

Following is a diagrammatic representation of different phases of waterfall model.

- Requirement Analysis
- System Analysis
- Implementation
- Verification
- Maintenance



Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are:

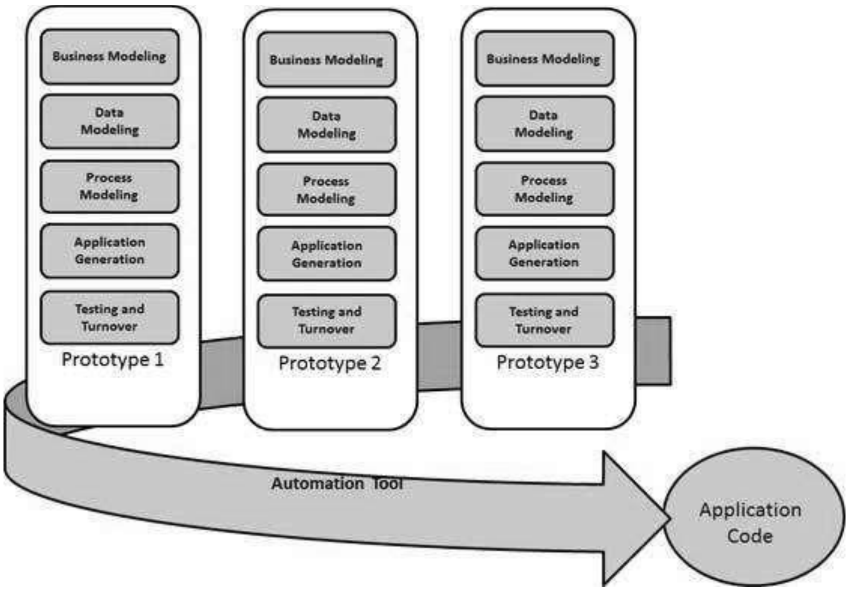
- Requirements are very well documented, clear and fixed
- Product definition is stable
- Technology is understood and is not dynamic
- There are no ambiguous requirements
- Ample resources with required expertise are available to
- support the product

Cons of Waterfall Model

- High amounts of risk and uncertainty
- It is difficult to measure progress within stages
- Cannot accommodate changing requirements
- Adjusting scope during the life cycle can end a project
- No working software is produced until late during the life cycle

Rapid Application Development (RAD) Model

The Rapid Application Development (RAD) model is based on prototyping and iterative development with no specific planning involved. The process of writing the software itself involves the planning required for developing the product. RAD focuses on gathering customer requirements through workshops or focus groups, early testing of the prototypes by the customer using iterative concept, reuse of the existing prototypes (components), continuous integration and rapid delivery.



Following are the typical scenarios where RAD can be used:

- RAD should be used only when a system can be modularized to be delivered in incremental manner.
- It should be used if there’s high availability of designers for modelling
- It should be used only if the budget permits use of automated code generating tools
- RAD SDLC model should be chosen only if domain experts are available with relevant business knowledge
- Should be used where the requirements change during the course of the project and working prototypes are to be presented to customer in small iterations of 2-3 months

Cons of RAD Model

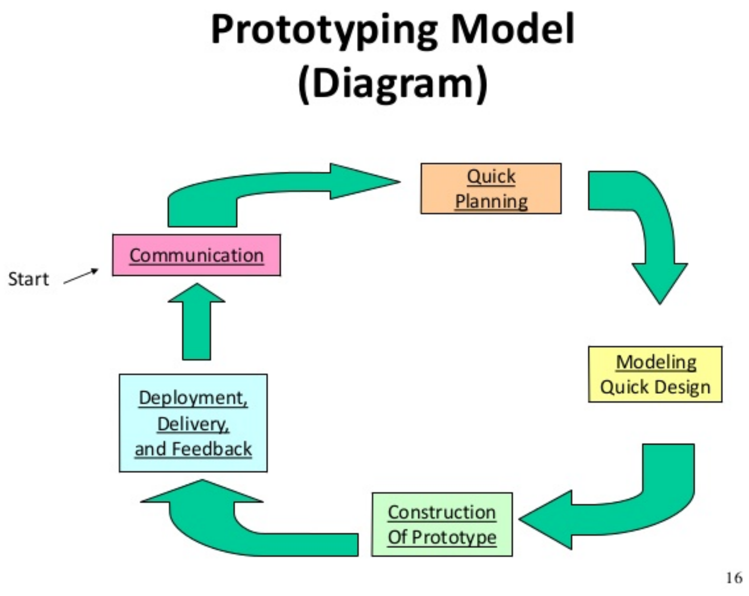
- Only system that can be modularized can be built using RAD.
- Requires highly skilled developers/designers
- High dependency on modelling skills
- Inapplicable to cheaper projects as cost of modelling and automated code generation is very high
- Requires user involvement throughout the life cycle

Prototype Model

The Software Prototyping refers to building software application prototypes which display the functionality of the product under development but may not actually hold the exact logic of the original software. Software prototyping is becoming very popular as a software development model, as it enables to understand customer requirements at an early stage of development. It helps get valuable feedback from the customer and helps software designers and developers understand about what exactly is expected from the product under development.

Prototype is a working model of software with some limited functionality. It also helps understand the requirements which are user specific and may not have been considered by the developer during product design.

Software Prototyping is most useful in development of systems having high level of user interactions such as online systems. Systems which need users to fill out forms or go through various screens before data is processed can use prototyping very effectively to give the exact look and feel even before the actual software is developed.


Prototyping Model (Diagram)
Start
Communication
Quick Planning
Modeling Quick Design
Construction Of Prototype
Deployment, Delivery, and Feedback
16

Cons of Prototype Model

- Risk of insufficient requirement analysis owing to too much dependency on prototype
- Users may get confused in the prototypes and actual systems
- Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans
- Developers may try to reuse the existing prototypes to build the actual system, even when it's not technically feasible
- The effort invested in building prototypes may be too much if not monitored properly.

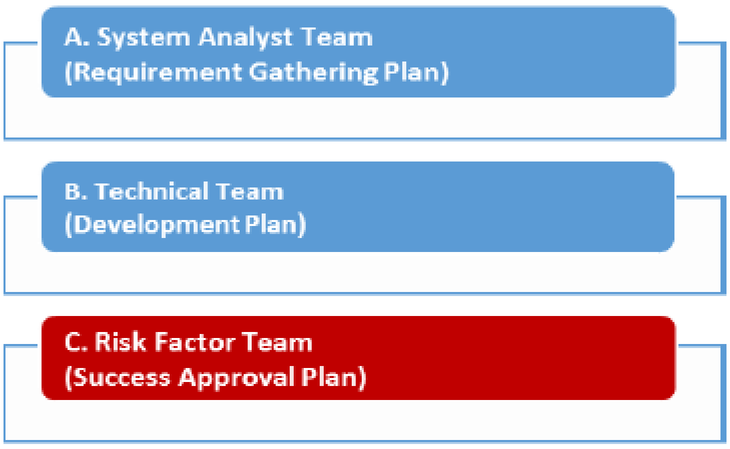
A1.2 Design of a software process model which suits the current and future software development requirements

NEW PROPOSED SDLC MODEL

The new SDLC model is planned in such a way that it allows software company and client to freely interact with each other in order to understand the requirements of software project in a good way to develop a good quality software within a given timeframe and budget.

SDLC process model start with the **client's requirements** so the proposed model tries to find every requirement like **functional requirements**, **non-functional requirements** and **user requirements** of the client/customer. It helps in developing a good quality of software product that satisfies the client/customer needs.

Finally, client satisfaction depends upon the good understanding about the client needs and associated user requirements for a better software product and the capability to connect those requirements to the software company. In addition, client satisfaction and confidence depends upon the level of product guarantee offered throughout the SDLC. Understanding to the requirements problems inevitably leads to poor client/customer and software company relationship, unnecessary re-work and exceed the budget and timeframe. The client satisfaction is totally depended on client needs for this reason the new SDLC model focus on the initial phases.



My proposed SDLC model include the following:

A. SYSTEM ANALYST TEAM (Requirement Gathering Plan)

The System Analyst team have a sufficient knowledge of computer science, software engineering, software development processes, software applications, operating system, as well as domain knowledge like various business functions to be performed. The system analyst team coordinates with the risk factor team and technical team. System Analyst team deals with the client for Identify Problem, Breakdown Requirements, make a Prototype, Finalize the Requirements, Feasibility Study, Approval of SRS and any ambiguity of client is also discussed and solved by the system analyst team.

System Analyst team works in following phases :

- Identify the Problem
- Identify the Requirements
- Breakdown Requirements
- Finalize the Requirements
- Feasibility Study
- Approval of SRS Document
- Make a Prototype

The system analyst team identifies the requirements and divide all the requirements into different features and then gets the existing software whose requirements match with the current proposed software requirements.

The system analyst team breakdown the requirements into two parts:

- Developed Requirement
- Non-Developed Requirements

Developed requirements are those requirements which are already implemented in some existing software.
Non-developed requirements are those requirements which are not implemented by any of the existing software, mean these are the fresh features and need to be developed.

After finalizing requirements, the system analyst team is now going quickly start work on Software Requirement Specification (SRS) Document and feasibility study report like estimates the budget, timeframe and effort which are mandatory for the development of the software product, as the SRS document is complete the system analyst team passes this SRS document (the final requirements document) to the technical team as well as to the risk factor team.

B. TECHNICAL TEAM (Development Plan)

Technical team is an expert team and its team members are updated with new technologies and new software products. It is a technically expert team. This team interacts with system analyst team during its working. Technical team studies the SRS document (the requirements document) received from the system analyst team which in turn get these requirements from the client/customer.

The member of technical team is full of skills and interacts with system analyst team. Technical team works on non-developed requirements. This team studies the feasibility of requirements to check whether these are technically possible or not. This team also identifies and resolves the various risk associated with the implementation of non-developed requirements with the collaboration of risk factor team.

After feasibility study and risk analysis the technical team finally verify the final SRS document, check the prototype provided to the client/customer and start work
on the following phases:

- Designing
- Coding
- Testing
- Implementation
- Feedback and Maintenance

C. RISK FACTOR TEAM (Success Approval Plan)

Risk are future uncertain events with a probability of occurrence and a potential for loss. Risk identification and management are the main concerns in every software project. Effective analysis of software risks will help to effective planning and assignments of work. Risks are identified, classified and managed before actual execution of program. These risks are classified in different Categories.

- Schedule Risk
- Budget Risk
- Operational Risks
- Technical risks
- Programmatic Risks

This risk factor team identifies the various risk associated with the requirements, so that the decision can be taken about the deployment/implementation of requirements. If some non-developed requirements are not technical possible then the client is informed about this during the early stage so that the client does accept the system with satisfaction and have no objection.

A1.3 Justification of the designed model with an Example

The following details explain how the new proposed SDLC model is applicable. The details given below explain how the new suggested Z-SDLC model has the command of satisfying the client/customer.

Software is developed for Company A, there are various SDLC models for the software development but I choose Waterfall model, Prototype Model and my new designed SDLC model for software development and comparing the working of existing models with the new SDLC.

A. Software Development by Waterfall model

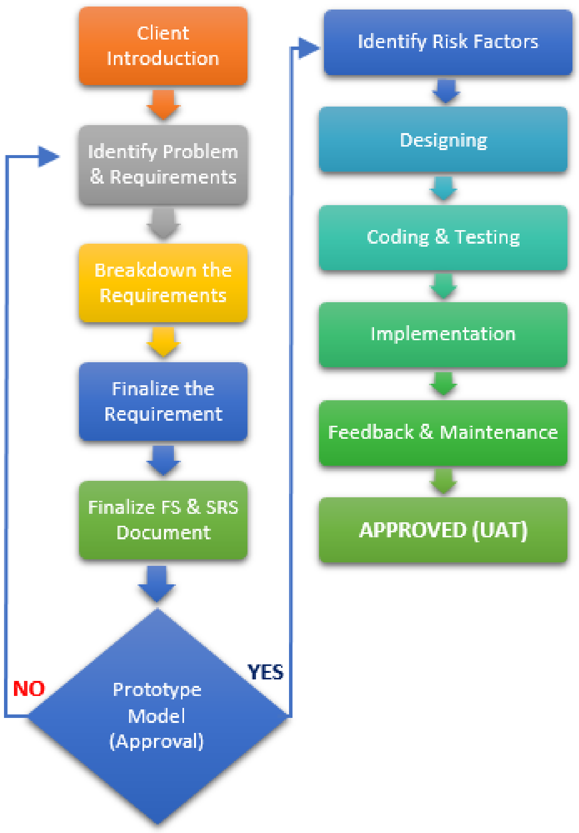
The waterfall model is a linear sequential model. We considered the requirements, check them and moved towards the designing phase followed by the coding and testing phases for the software. This was not accepted by the client because they were not satisfied. As the client want to change it in terms of graphics, functionality and features the waterfall model does not allow to change after completing the requirements so, it fails to convey client about the software product.

B. Software Development by Prototype model

The prototype model build prototype to give feel of the proposed software to the client. As we already have requirement so, we build prototype and showed it to the client. After client's feedback, we changed it and again showed it to the client. After building and showing three prototypes, client finalized the requirements and we passed these final requirements to next phases for software development. Finally, the software was delivered to the client. But building prototype affects cost, schedule and effort which get exceeded.

C. Software Development by new SDLC model

The striking feature of this model is the client satisfaction. Firstly, system analyst team deal with the client to discover the problem and requirements. After discovering the problem, the system analyst team breakdown this requirement into developed requirements and non-developed requirements.
Accordingly, the system analyst team now breakdown the requirements into developed and non-developed requirements. The system analyst team finalize the requirement and also start work on the feasibility study and SRS document.
The system analyst team with the collaboration of technical team analysed the available requirements provided by the system analyst that is present in SRS document for the
proposed system and searched the most matching software for them. He found three such software whose requirements matched with the proposed software's requirements previously build for another company.



Now the system analyst team showed the software to the client so that the client got the feel of proposed software and also give his suggestion and feedback to the system analyst team. The system analyst team again passed these suggestions to the technical team and the process goes on until the client finalized the requirements. System analyst team passed final requirements to the risk factor team for the risk analysis and the requirement validation.

After validation and resolving various risk associated with the final requirements, these final requirements were passed to technical team for final software product.

The technical team start work on the following phases i.e. designing, coding, testing, implementation and maintenance followed by the validation process to develop the final product.
The final product was approved by the client because it satisfied the client's requirements within budget and given timeframe because budget and timeframe were not disturbed or affected due to various increments or by building prototype. Finally, User Acceptance Test is signed with the client.

Bibliography

- https://www.roberthalf.com.au/blog/employers/6-basic-sdlc-methodologies-which-one-best
- https://www.tutorialspoint.com/sdlc/sdlc_quick_guide.htm
- https://pdfs.semanticscholar.org/6c66/61ab8ae91a59c28803bcae7e6d63eadfc904.pdf
- http://www.visionraval.com/list-of-software-development-life-cycle-models-hybrid-model/
- https://content.intland.com/blog/agile/when-why-how-to-use-the-hybrid-model
- https://geeksforgeeks.org