DESIGN AND DEVELOP CONCEPTUAL INTEGRATED PROCESS MANAGEMENT METAMODEL USING AGILE - HUMAN CENTRIC AND DESIGN THINKING APPROACHES

A Thesis Submitted

IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY
IN
COMPUTER SCIENCE & ENGINEERING

By

RAJEEV SHARMA
Reg No. 18SCSE3010011

Supervisor
Dr. J.N. SINGH
Professor

SCHOOL OF COMPUTING SCIENCE & ENGINEERING
GALGOTIAS UNIVERSITY
UTTAR PRADESH

December-2023

APPROVAL SHEET

The Ph.D. thesis entitled **“Design And Develop Conceptual Integrated Process Management Metamodel Using Agile – Human Centric And Design Thinking Approaches”** by **Rajeev Sharma** is approved for the degree of Doctor of Philosophy in Computer Science & Engineering.

Examiners

____

Supervisor (s)

Prof. (Dr) J.N. SINGH

Chairman

____

Date:__________

Place:__________

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis, entitled **"Design And Develop Conceptual Integrated Process Management Metamodel Using Agile – Human Centric And Design Thinking Approaches"** in fulfillment of the requirements for the award of the degree of Doctor of Philosophy in Computer Science & Engineering and submitted in Galgotias University, Uttar Pradesh is an authentic record of my own work carried out during a period from September-2018under the supervision of **Dr. J. N. SINGH,** Professor**,** School of Computing Science & Engineering, Galgotias University.

The matter embodied in this thesis has not been submitted by me for the award of any other degree or from any other University/Institute.

Mr.RAJEEV SHARMA
18SCSE3010011

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Dr. J. N. Singh
Supervisor
SCSE
School of Computing Science & Engineering
Galgotias University

The Ph.D. Viva-Voice examination of Rajeev sharma, has been held on ______________

.

Sign. of Supervisor(s) Sign. of External Examiner

ABSTRACT

In this research, the researcher, design and developed conceptual integrated process management metamodel that improves the human centred software development (HCSD). Human centred software development (HCSD) is approach of software development for design and develop according to end-users'requirements and feedback. In this reserch, the researcher design and developed conceptual software process management metamodel by the integration of agile-human centric methodologies like : agile extreme programming (XP), scrum technology with the design thinking (DT) approaches. This metamodel increase process management, response of end-users's feedback and requirments, efficiently managed the users's incoming data, and the amount of data it can be quickly capture and passed for further development stages of software development life cycle (SDLC). This research employs that developed conceptual metamodel provides development team an environment of out of the box thinking for problem solving on the basis of end-user's requirements and demands. Agile engineering approach is an enabler to accelerate software product delivery on time, manage user's priorities in the dynamic stages of software designing and increased the product efficiency and effectiveness. Design thinking (DT) approach provide quality and potential solutions for product and services and an increase productivity and end-user's operational behavior improvement and also takes the advantage of innovation ideas in ongoing process delivery and increase user acceptance index factors in every stages of software development designing and process management in context to human-centric environment. Develop conceptual human centred software development (HCSD) metamodel main working domain is capture dynamic requirements of end-users in proactive manner and also produce systemtatically and realistic solution towards the changeable environment of business stakeholder and designing team feeeback and assurance of product delivery. This huma centred software development (HCSD) metamodel increase statisfaction of designing phase and analyse their user statifaction level through the product reliability metrics and by the end-users feedback and provide the conceptual solution for the end-users requirements prospective. This developed conceptual metamodel provide synthesis and validate solution by the involvement of users in through out the life cycle of process designing and development and gives assurance towards the designing goals and requirements. This develop conceptual metamodel provide feedback and response interface throughout the designing and development phases of software and process management and for this approach designing team and stakeholder takes the end-users responses/requirements in dynamic stages of process re-engineering and development and also involved data predication and forecasting for data realiability. The human centred software development (HCSD) metamodel works optimaztions of clients requirments and reduce the gaps between the development team members communication and make data retention of user’s statisfaction on top priority at every stages of software development and process designing and also increae the user acceptance index factors and standards through data integrity and validation process and activity.

Keywords: Human Centred Software Development, Process Management, Agile Engineering, Design Thinking (DT) and Product Reliability.

ACKNOWLEDGEMENT

Working as an Assistant Professor and doing research for the degree of Ph.D. in Galgotias University, was quite magnificent and challenging experience for me. In all these years, many people directly or indirectly contributed in shaping up my career. It was hardly possible for me to complete my doctoral work without the precious and invaluable support of these personalities. I would like to give my tribute from bottom of heart to all those people.

Initially, I would express my sincere gratitude to my supervisor, **Dr. J. N. Singh, Professor,** School of Computing Science & Engineering, Galgotias University, for his valuable guidance, enthusiasm and overfriendly nature that helped me a lot to complete my research work, for his motivation, feedback and suggestion on research questions, experiments and writing and presentations that followed, philosophical discussions, and vithe wisdom and knowledge conveyed. I express my sincerely thanks and gratitude from bottom of heart to Shri Sunil Galgotia Chancellor and Dr. Dhruv Galgotia CEO, Galgotias University for giving me the opportunity to carry out my doctoral research and for their support. My special thanks to Vice Chancellar Dr. K. Mallikharjuna Babu, Registrar Dr. Nitin Kumar Gaur, Pro Vice Chancellar Dr Avadhesh Kumar, Ms. Aradhana Galgotia, Hon'ble Director Operations, Prof. Renu Luthra, Advisor to the Hon'ble Chancellor, Dean Acedamic and Dean SCSE Dr. Sanjay Goel, Dean of R & D, Director-HR, Research Coordinator SCSE, Dr. Sunita Yadav Associate Dean SCSE, COE Dr. Bhattacharya, Professor Unni Krishnan and Dean University Polytecnhnic, for their continuous help and support to accomplish this research. Due to their excellent brilliant ideas, able guidance, motivations and directions, I could grow as a Ph.D. scholar and produce this work and my sincere thanks to SCSE faculties and staff membes for their support and suggestions to accomplished this research. I give my thanks and regards to Dr. Subhash C. Sharma, Professor, Electronics & Computer Engg, IIT Roorkee.

I heartily thanks to my family members for giving me adequate time and support to complete this research because this research work not be completed without their support and motivation and I would give my thanks from bottom of my heart to GOD for his bliss and grace always in every stages of my life. I give my parnam to my mother and father on this stage of my life and my father specially because he is no more in my physical life but his grace and love from the brigher world, I always experienced in my life.

(Rajeev Sharma)

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

Abbreviations	Descriptions
DT	Design Thinking
XP	Extreme Programming
HCSD	Human Centred Software Development
EJB	Enterprise Java Bean
DCOM	Distributed Component Object Model
SDLC	Software Development Life Cycle
RUP	Rational Unified Process
RAD	Rapid Application Development
AML	Agile Modeling Language
FDD	Feature Driven Development
MVP	Minimum Viable Product
SD	Service Design
Loc	Line of Code
SSM	Structure Synthensis Model
JAD	Joint Application Development
SRsD	Systematic Literature Reviews Document
OSSD	Open Source Software Development
KLoC	Thousands of Line of Codes
SDL	Service Domiant Logic
ASD	Adaptive Software Development

DSDM	**Dynamic System Development Method**
AM	**Agile Modeling**
ASP	**Agile Software Process**
PP	**Pragmatic Programming**
DSDM	**Dynamic System Development Methodology**
SQA	**Software Quality Assurance**
FDD	**Feature-Driven Development**
SAFBE	**Scaled Agile Framework for Business Envrionment**
MDRE	**Model Driven Reserve Engineering**
HCTILS	**Human-Centred Technology-Independent Level of Specification**
HCSM	**Human-Centric Structure Metamodel**
CSMM	**Conceptual Structure Metrics Metamodel**
VV	**Verification and Validation**
AESD	**Agile Engineering Software Development**

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHM

LIST OF PUBLICATIONS

International Journal:

1. Rajeev Sharma, J.N Singh, "An Intelligent Human Centred Software Development Framework" Journal Name (International Journal of Information Science and Applied Engineering (IJISAE). Scopus), ISSN: 21476799, Vol 11 NO. 10S (2023).
2. Design-Thinking and User's Requirement Engineering: Human Centred Development, International Journal of Mechanical Engineering, ISSN: 0974-5823, Vol. 6 (Special Issue, Nov.-Dec.2021) Rajeev Sharma[1], Jitendra Nath Singh[2], Galgotias University, Greater Noida, Uttar Pradesh, India. (Scopus International Journal)
3. Rajeev sharma, J.N. Singh, A Critical Review of Surveys Emphasizing on Agile software engineering, solid-state journal ISSN:0038-111X Vol. 64.2(Oct-2021) (Scopus International Journal)

International Conferences:

1. Human Centre Software Development Model and User Acceptance Index Factors Count, International Conference in Multidisciplinary Concepts in Management, ICMCM-2023, Glbjaj Greater Noida, June 30 & 1 July 2023.
2. Systematic Literature Review and Comparative Studies on Human Centred Software Development, Rajeev Sharma,& J.N.Singh, 16th-17thDec.2022,4th IEE International Conference on Advances in Computing, Communication Control and Networking,2022 (ICAC3N-22).
3. Human Centred Software Development Approaches, Rajeev Sharma, & J.N. Singh.,17th-18th Dec.2021, 3rd IEEE International Conference on Advancesin Computing,Communication Control and Networking,2022 (ICAC3N-21).
4. Human Centre Software Development Model and User Acceptance Index Factors Count accepted for the publication in World Scientific Publishing.

Patent:

1. One Patent is done under Research Guide: "GGU-006-067 || GGU-001-071" and Patent Application No: 202111053463.

CHAPTER - 1

INTRODUCTION

INTRODUCTION

In the last few years, fast development growth and adopts new ideas, thoughts, innovation by software development industry, and parallelly enhancement in software designing and development methodologies taken attention from product delivery context to product quality and end-users' requirements satisfaction and point of view, so the current software development come across the "human centred software designing and development" [1]. Now days, in software development industries end-user's involvement brings in every stages of process designing and development and often provides continues verification and validation in every phase of process designing and development so the software development now switched to traditional process/product development to human centred software development [2]. Human centred software development approach is currently one of the most emergent, emphasize, and standard approaches capable to increase end-user effectiveness by ensuring the compatibility between end-users' requirements satisfaction and demands and achieved the goal and objectives of the stake holders and business prospective from the software development methodologies and delivery those decided in initial stage of software development [1-3].

The evolution of the software process designing and development now moves towards "static and rigid theoretical approach of service design and development to the next evolutionary environment of software development methodologies and techniques" and mainly emphasis on involvement of new and advance thoughts and ideas of software designing, creativity, and assurance of end-users' thoughts and their expectations from the software development and also critically validated and advancement growth possible and overcome the limitations of process development [4]. Software project development is a process development life cycle (PDLC) directly integrated and given a framework to manage and control the real end-user's requirements satisfaction and overcome the project complexity toward the project handling and implementation [5]. Agile engineering methodologies mainly focus on efficient and effective delivery of process and product towards the end-user requirements and stakeholder point of view and in short time span frame and rapid and changeable development growth environment [6]. Design thinking approach work started as early as possible in software development life cycle (SDLC) by assuming the users' mindset and expectation from the designing and development team and leaves their observation, methodologies, thoughts, logics, and experiences behind when going to fixing the client/users' assumptions and requirements to develop software and application according to them and their point of view [7]. Human-centred software development (HCSD) approach is not possible without fast execution of requirements and process designing according them and finally product delivery to the end-users and stake holders comes across in effectively and efficient manner and also achieved the standards and quality [1-2]. So for this purpose we proposed, a conceptual integrated process management metamodel using agile-human centric and design thinking approaches" which we discuss in this thesis. A software development methodology are set of some predefined rules and regulations that are used to designed and develop software from first phase requirement specification to post implementation phase and so this reason software development fallow and work around some phases and techniques [10-13]. These methodologies and stage become integrated core essential part of software development life cycle (SDLC) to designed and developed standard and quality software and product for end-user requirements satisfaction and objectives, and in "current and modern software development era, these strategies sometime fix and dynamic and also work form linear to non-linear way of software development and it's all depend on the end-users' requirements and needs" [10-15], some of them software development methodologies are discussing below:

1.1. TRADITIONAL SOFTWARE DEVELOPMENT (TSD) APPROACHES :

Traditional software development methodology, fallow some pre-defined rules and guidelines that are used to developed software from some different phases e.g., client requirement and end-users' feedback, requirement analysis, planning, designing, code and deployment, testing, implementation and post implementation operational and maintenance phase [8,10,11] and in brief traditional software development approaches (TSDA) fallow predefined static software development life cycle (SDLC) phases and techniques to develop software process and product and in software development industry traditional software development approach commonly come under the umbrella of "Water Fall Software Development Model" and it has generally five phases and it is called software development industry first development model and first formally introduced as a method for software development in an article written by Winston W. Royce in 1970

[8,13-15]. This traditional software development approach is called “Water Fall Model” now days.

As shown in below figure 1.1 traditional software development model has five phases, requirement analysis, designing, code implementation, testing and validation, and last phase is called maintenance and post implementation phase [1,8].Traditional software development generally adopted five phases of software designing and development e.g. requirements analysis, designing, coding, testing and implementation [2] and fallow sequential level of process designing and development in entire software development life cycle(SDLC).Traditional software development approaches reduce the process designing scheduling time and cost and in other end by these approaches reduce the chances of issues and bug in software development if the requirement might be fix and rigid but if requirements are dynamic and open types then productivity will affected and chancesof issues and complexity increased and sometimes very difficult to work on them for designing team.

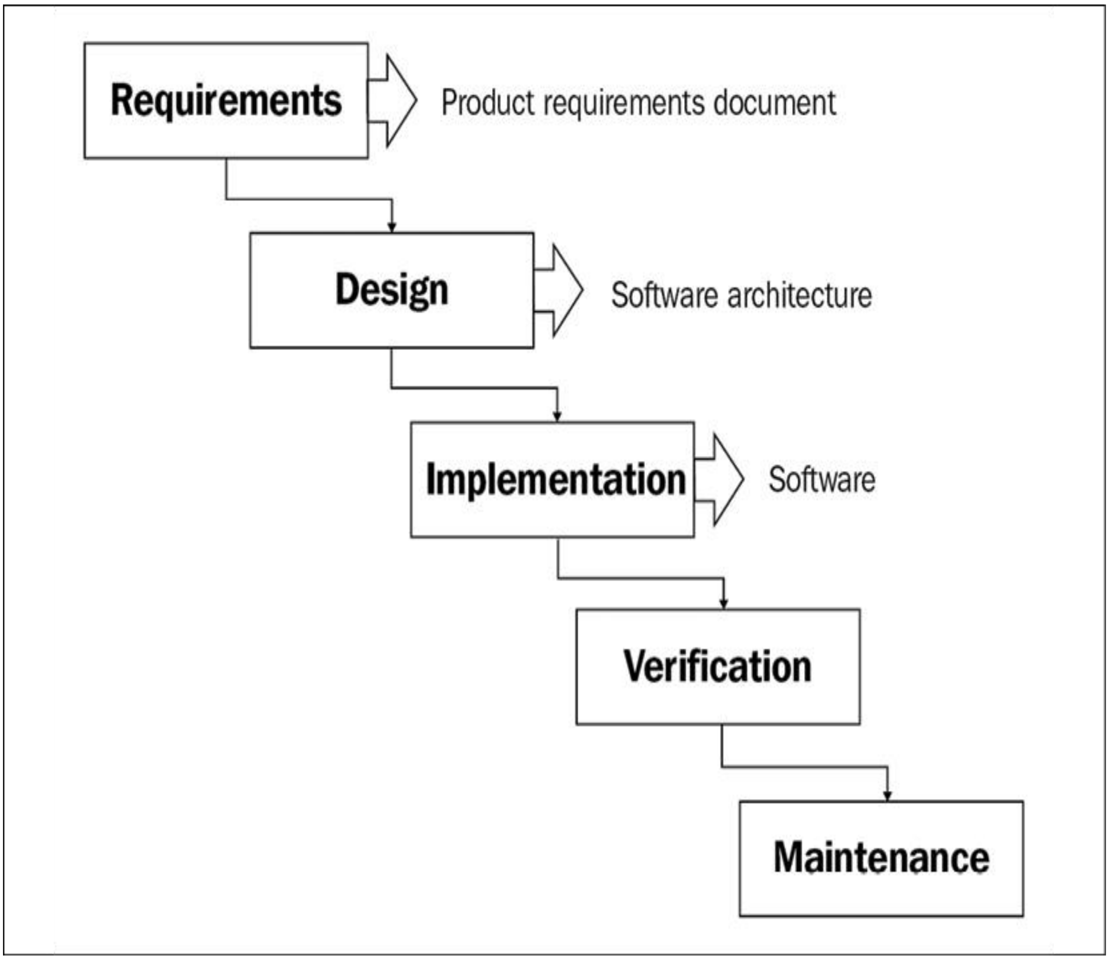


Figure 1.1. Traditional software development approach

1.1.1. Water Fall Software Development Model Features:

Water fall software development methodology work on one-way development methodology. Given below we point out some features of traditional water fall software development model features:

- Detailed and well-defined requirements
- Requirement must be static not be dynamic
- Product clarity should be static
- Project size is fix and small
- Possibility of ambiguous requirements very rare.
- Used well defined technology and methods for system designing
- Minimal time required for project development

1.1.2. Spiral Model :

Spiral Model is first model of software development industry that involved risk analysis and risk factors in software designing from the very beginning phase of software development life cycle (SDLC), basically this model was introduced firstly in software industry by 'Boehm'. Spiral software development model used mainly to developed large, complex and expensive projects and software and apply both the approaches of software development bottom to top and top to bottom to develop software and system in very short and minimal time period [9]. Spiral model core area of software development is come around to the identify the risk and find the best and secure ways to resolve risks from the software development life cycle (SDLC) [4,9]. In given below some of the important features of spiral model are listed:

- Working on objectives determination and finalization
- Provide alternative solutions and fix the risks
- Provide next version of product solution
- Monitoring, review for next stage of software developmentation
- End-users involvement started from first phase of software developmentation
- Chances of issue and bugs are very less in compare to traditional methods of software developmentation

- Always concern about next level of software developmentation
- Verification and validation process run on software designing process and software quality assurance in parallelly

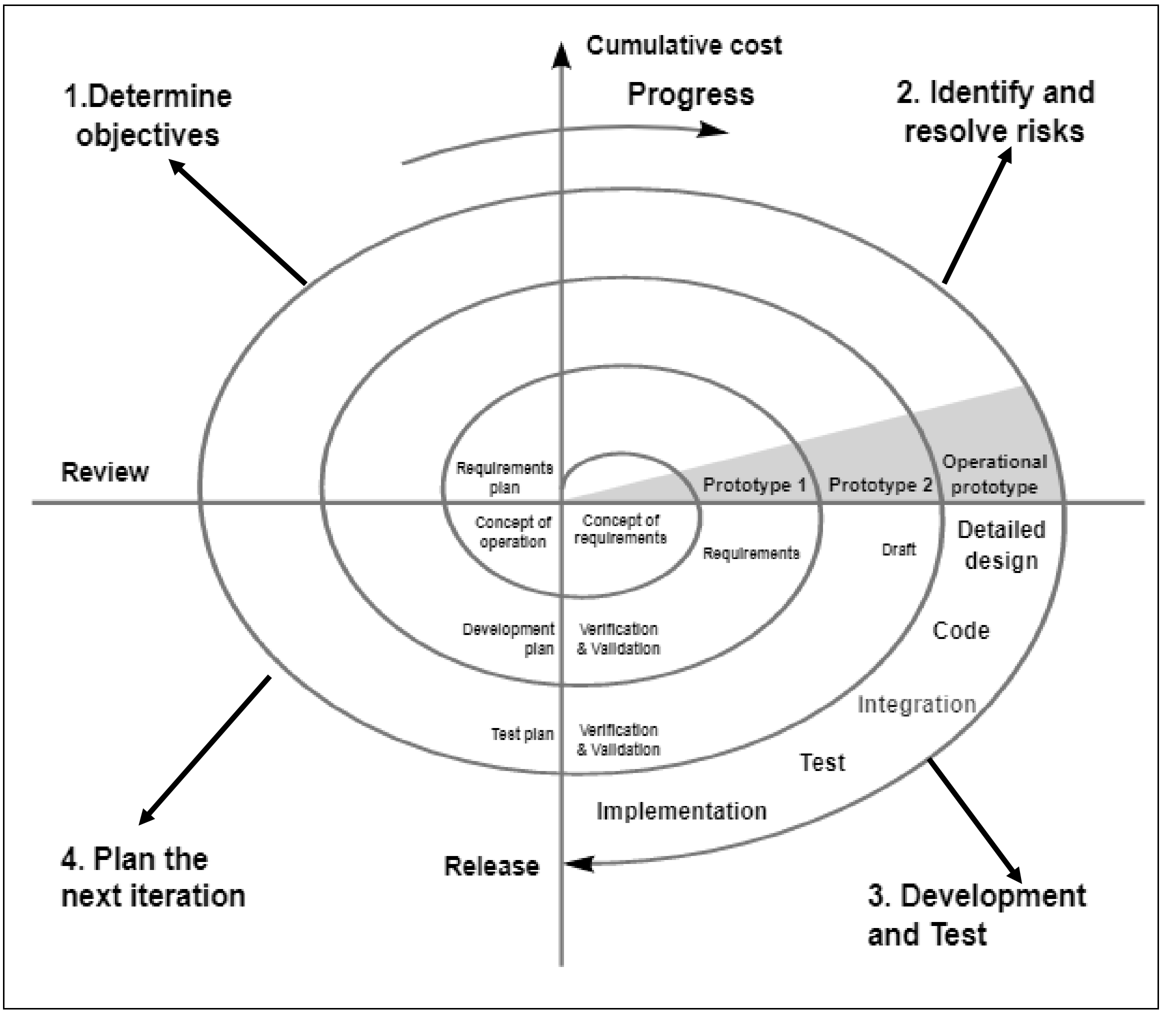


Figure 1.2. Spiral model of software development [10]

Spiral model different phases of software development and their working flow are shown in above figure 1.2. The following points give a brief knowledge and view about the spiral model of software development [4,10-15]:

1.Planning for objectives: This phase main work is understanding of software process requirements by applying the continuous point to point communication between end-users and with the planning team of resources analysts.

2. Risk Analysis Phase: This is most effective and core phase of spiral model, in here we decided the risk and also applying some authentic and efficient techniques to resolve the issues and risk factors from the development phase.

3. Development Phase: In this phase developmentation and testing running parallelly to complete the developmentation of product or application.

4. Plan to Evaluation Phase: In here, allow to real users and stakeholders to validate the development according to their end needs and demands before next incrementation phase of software developmentation begin.

1.1.3. View Model of Software Development:

In software development V model is used when development and testing required parallelly in planned manner to developed software. So, in conceptual level “verification and validation” both the strategy applies at every phase on same time of process designing. View model used “high level design (HLD) and low-level design (LLD)” to achieved the standard and quality software development. In HLD approach system is broken down in one or more than one sub module to getting multiple functionality and designing prospective regarding system designing and in LLD technique is used to verified and validated each phase independently and confirm that design is compatible with internal and external sub system in architectural and in other aspects of software designing. Given below, we discus some core attributes of view model (V-Model) of software engineering:

- Requirements specification are well declared, clearly documented and fixed
- Software product definition is understood and static
- Methods and technology used in software development is well understandable by software designing team members and not used dynamic approach in meanwhile of software development and designing
- No undefined and ambiguous approach used for software development
- View model development strategy suitable where the length of project is short
- View model software development methodology generally uses where quality assurance standard required on priority basis
- By the use of view model software development product cost may be reduce and increase the efficiency and effectiveness of software process delivery
- Dynamic requirements criteria and adoption is also open in this approach

View model enhance the product credibility, portability, operation ability, quality assurance, user satisfactions as far as possible in software development and given below figure 1.3 shown the framework and different phases of view model and their working flow.In view model client requirements specification designing process and process architecture process implementation run parallely to increase the effectiveness and efficiency in software development process and in last view model release working platform for system prototype and architectural enivrionment according to end-users requirments specifications.

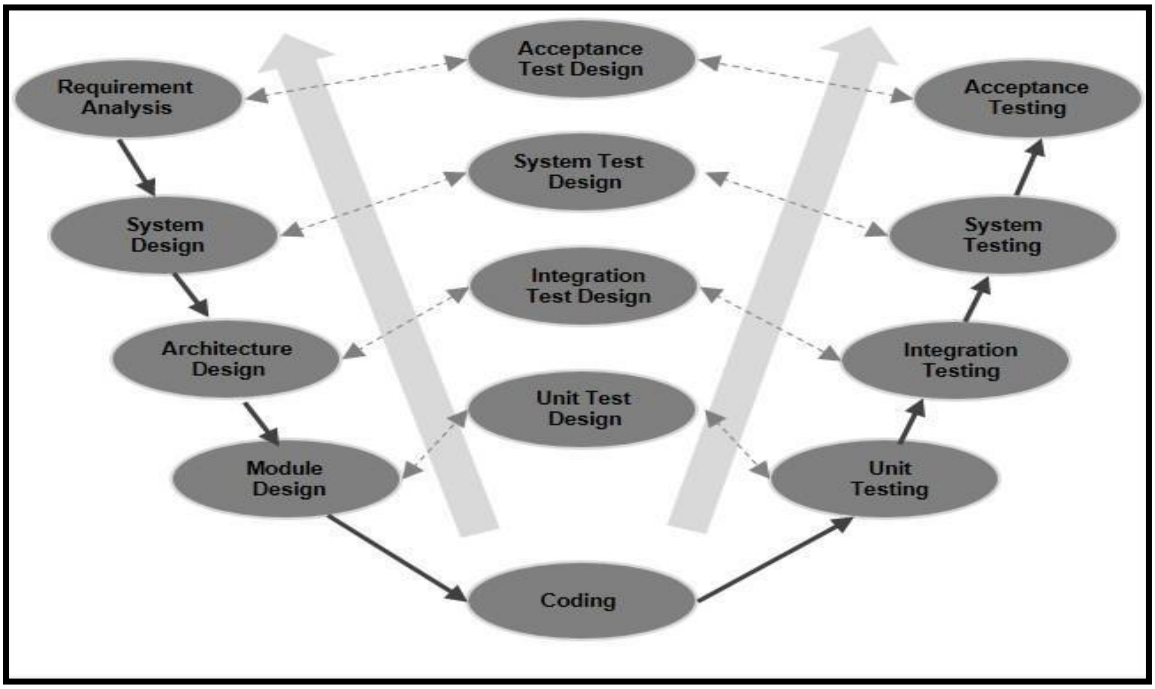


Figure 1.3. View (V) model framework and phases

1.1.4. Incremental Model:

This model used the inherit property and attribute of “software development water fall model” in more dynamic and iterative manner. This model emphasizes on each and every segment of process and produces sequential and linear versions of product and application in an incremental way. By this model we firstly implement the core part of product and process and then by the users requirements and feedback add more functionality and behavior in running product and process to improve the sequential construct product and until the all decided functionalities and requirements are deployed successfully and get the expected environment and result from the developed software.

Product according to users and stake holders and concurrent activities of increment model for software development shown in below figure1.4.

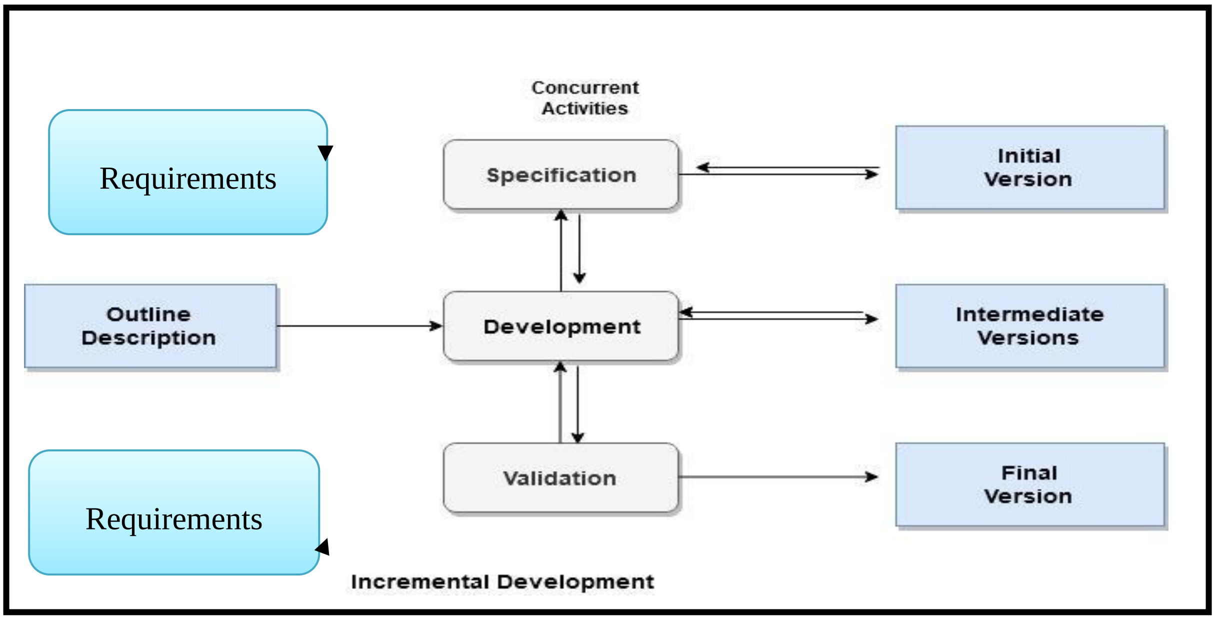


Figure 1.4. Increment model for software development

Given below point out some attributes of incremental model:

- Incremental Model is also called "Iterative Model of software engineering"
- Customer realize that deliver system functionality as earlier in development phase of software development and if any updations required then updated to software development team
- If pre-defined functionality not capture in current focus then development team try to come in down in next increment phase of product
- development. Increment model mainly focus on specification, process development and validation of end-results and product.
- Increment model core focus area in communication and feedback within all phases of software development.
- Incremental software development model has flexible approach to development and the end-product easily to implement.
- Incremental software development increase the dynamic satisfaction through involved end-users requirments in realistics manners.
- Incremental software development work on advanced sophisticated development style to increase efficiency and effectiveness of process designing and development.

1.1.5. Fountain Software Development Model:

This software development approach is based on user oriented/driven and object based and driven and fundamentally it is suitable for object-oriented software designing and development. This software development model has no predefined strategy and specific order of approach for process designing and no clear boundaries of software development phases and fountain model core advantage is that if some progressive part and unit is missing in other phase of software development then it will be added in software development life cycle (SDLC) by the incremental or iterative software designing approach [10,16] and below figure 1.5 shown the different phases of fountain software development model and also showing that all the development phases in fountain software development model are in open edges to predict and catch the requirements from the user level.

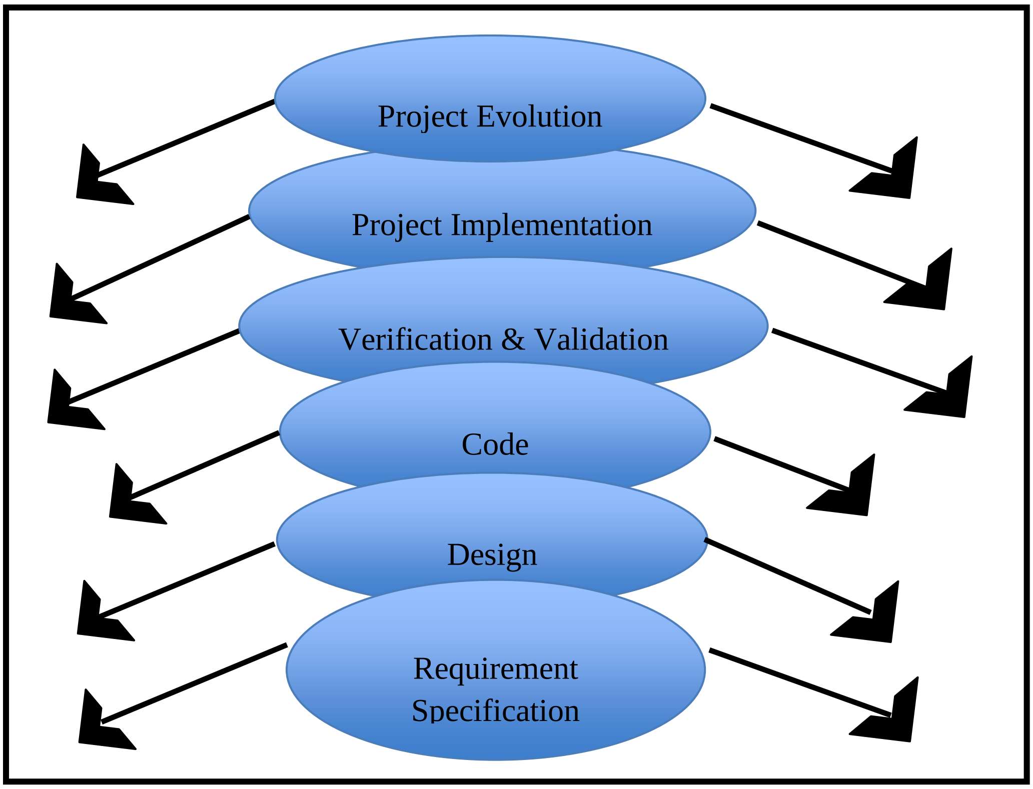


Figure 1.5. Software development fountain model [16]

1.1.6. Prototype Model:

Prototype development model basically is integrated cluster of decisions that point out and dictate what type of action taken place to solve problems and how accomplished the development process [17]. Prototype software development model provide platform that commit for business interest and software engineering practice too and for this prototype software development model approach takes feedback by customer, cost analysis and scheduling of project and other end prototype model design conceptual base model for software designing and expand the designing quality in future too [16,17]. Prototype development approach work on both edges of software development: functionality of system and user interface level of system and prototype model also provide future driven software development approach and capture the optimize requirements specification and acceptance of end -users [18] and below Figure 1.6 shown different phases of prototype software development model and showing that work start from requirement analysis phase to user feedback and in last after the binding and evaluation stages final product release.

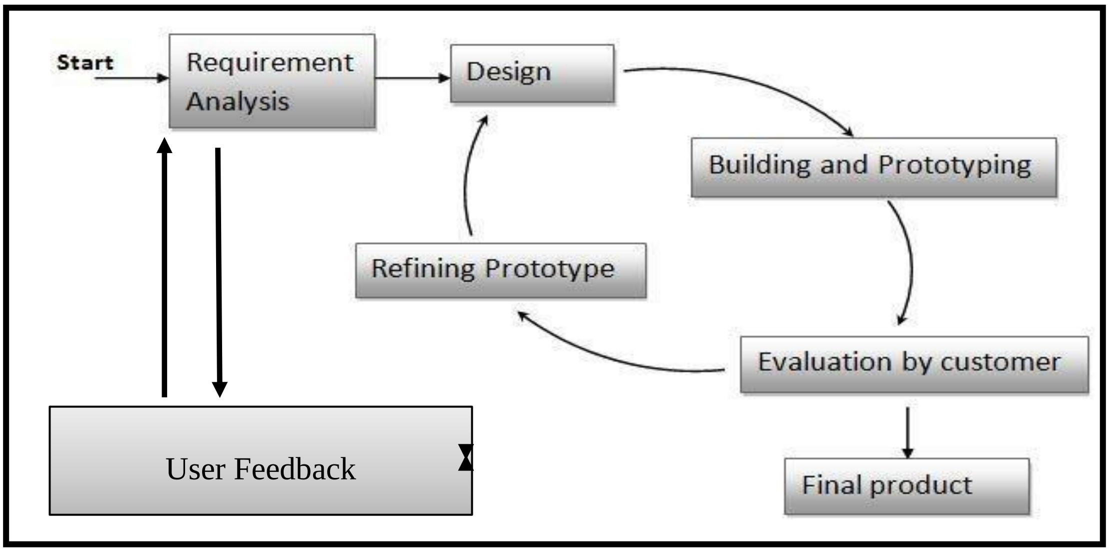


Figure 1.6. Prototype software development model

1.1.7. Winwin Software Development Approach:

In this approach we integrated spiral model with prototype to point out risk analysis and identification and by the earlier feedback and negotiations with users and stakeholders to ensures and achieve an objective decided by users and software development activities [19]. Winwin software development approach gives advantage to both development team and end-users regarding the emerging risk factors in every stages of software designing and parallelly resolved and provides responds accordingly in evolution stages of software development [20]. Winwin software approach one hand, decided functionality of software designing according to the users' requirements and specifications and other hand, give platform and methods to developer to finish tasks and work within budget and given time frame and resources [20]. Different phases and working environment of winwin model shown in givne below figure 1.7.Winwin software development model has some following features:

- Integration of Spiral model and Incremental model of software development and to enhance designing and quality of product.
- Reduce the cost, time of the project and develop product within time frame
- Develop product within time frame and schedule
- Deployed dynamic approaches of software development to maintain technical integrity of the software development life cycle different phases
- Verification and validation play significant impact in software development
- Reduce the complexity in software designing process and coding by adoption of effective and feasible approaches of software methodologies
- User level integration always on high priority

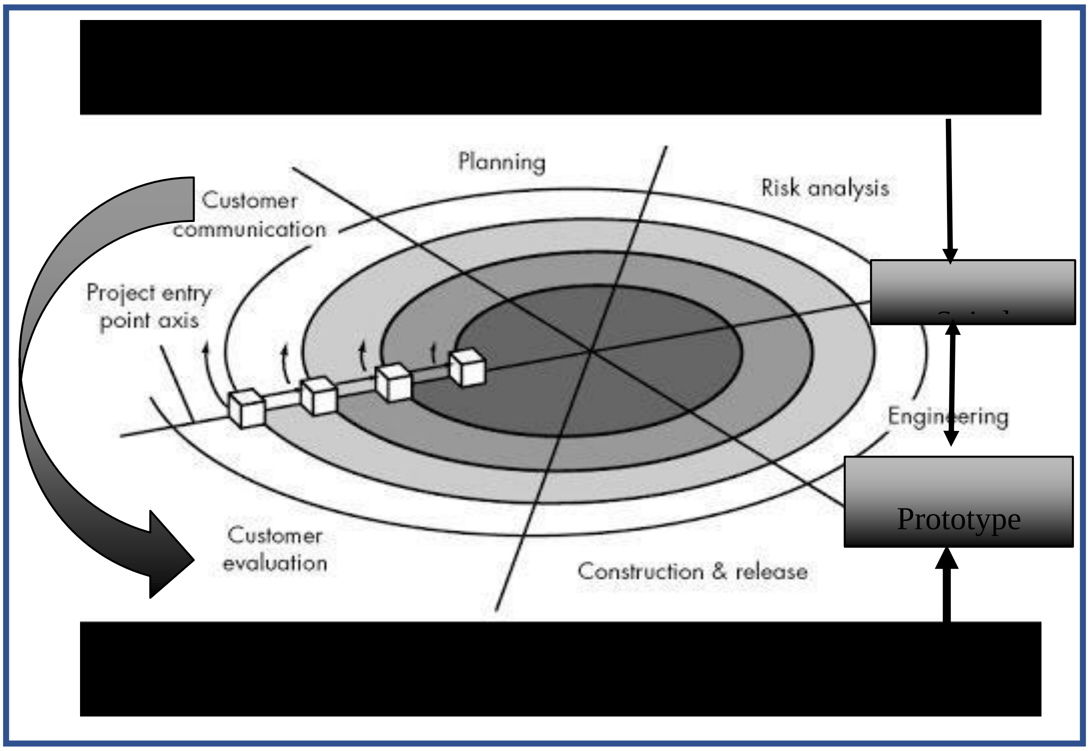


Figure 1.7. Winwin software development model

1.1.8. Intelligent Software Development Model:

In this software development approach water fall model approach combines with expert system, so this model also called “knowledge base software development model” because this software approach workout on information-based methodologies for software development life cycle (SDLC). Intelligent software development model basically called “information based expert system for software development and product.” This development model used set of functionality and tools to design data query, reporting, data and information processing, form designing, user acceptance level of front end designing and codes, graphical representation of methods and data flow for high level software and system designing and also used expert system knowledge based experience to developed system [21] and below Figure 1.8 showing the life cycle of knowledge based intelligent expert system and data flow in each pahses and knowledge transformation also.

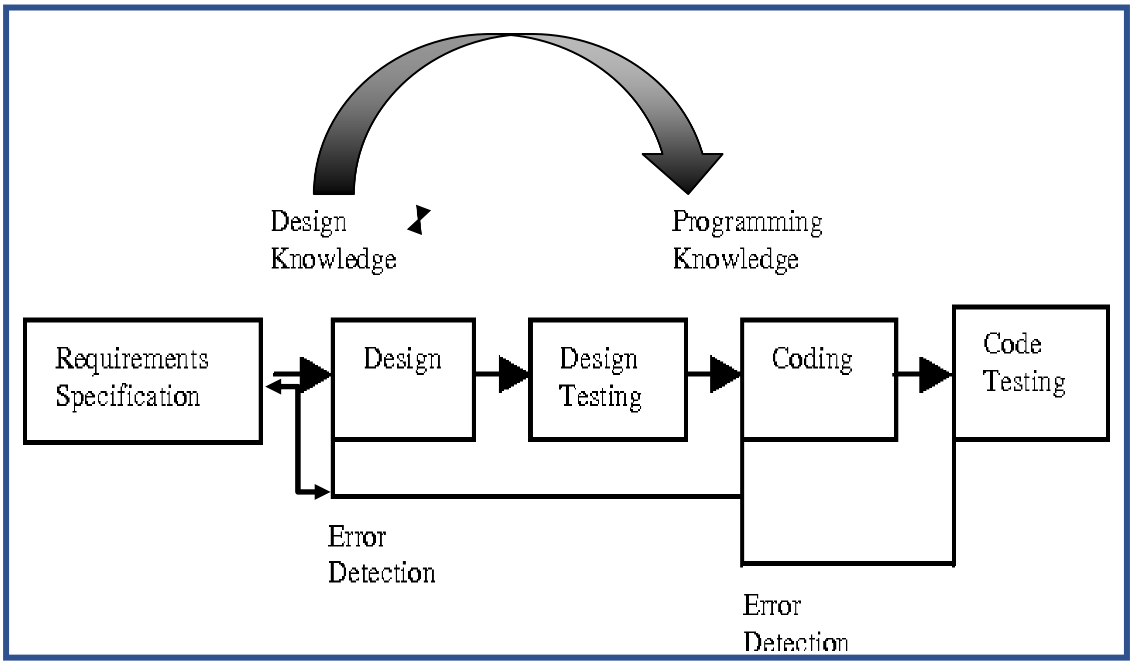


Figure 1.8. Knowledge based intelligent expert system

1.1.9. Parallel Software Development Model:

Parallel software development based on parallel software designing methodology and focus on state of technical requirements and series of technical methods and task to develop activity-based networks in software development life cycle (SDLC). This software the development model points out precisely and monitors every activity running on different environments and provides a correct and accurate image of software designing within different software development states of the project [19]. In this approach verification and validation of process desiging and product implementation run in paralled manner to gives the product assurance and effectiveness. Below Figure 1.9 shown the parallel software development approach different phases and their working environment.

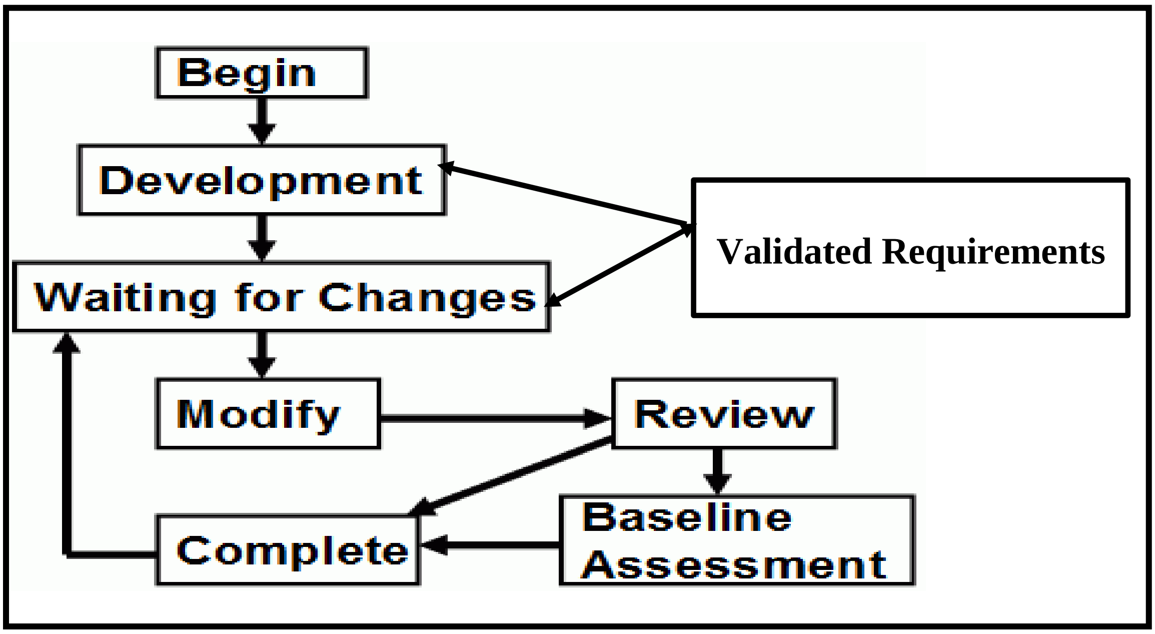


Figure 1.9. Parallel software development approach

1.1.10. Architecture-Based Software Development Model:

This model conceptually based on architectural framework and component-based software designing approach and practices and this software development conceptually depend on iterative approach of software development and mapping and analysis the functional dependencies with structural process design phases of software designing and development and in this development model requirements acquisition and their analysis paly significant role to provide better solution and framework to develop software and product in standard manner [19, 22-23]. Architecture - based software development model integrated the logic view, process view, development view, and physical view in conceptual manner to provide object oriented solution and subsystem based designing environment and process development.

This architecture-based software development model core focus on conceptually designing methodologies and their impact level dependencies in different edges of software development life cycle (SDLC) phases as shown in below figure 1.10. Architecture-based software development model takes the features of user-driven practices putting together in object-oriented decomposition and subsystem decomposition in simultaneous manner to release physical view of software product.

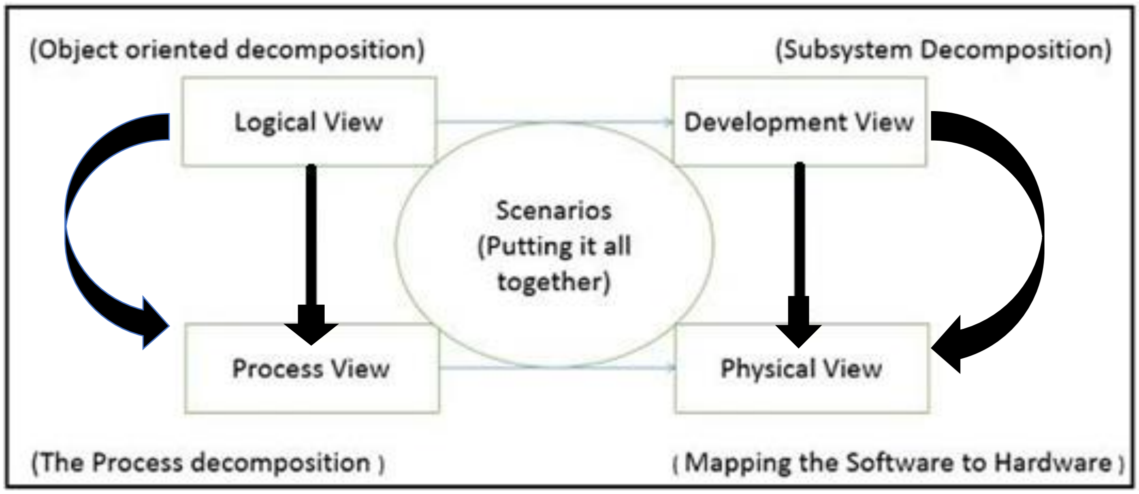

(Object oriented decomposition)
(Subsystem Decomposition)
Logical View
Development View
Scenarios (Putting it all together)
Process View
Physical View
(The Process decomposition)
(Mapping the Software to Hardware)

Figure 1.10. Architecture based software development approach

1.1.11. Component-Based Software Development Model:

In component-based software development approach different components based on their requirements and demand of rapidly development of software process specially in software designing tools e.g. EJB, DCOM, and product-based tools designing and development environment and most of modern software industry technologies used and focus on component-based software development to enhance the productivity and efficiency of software designing and also increase the quality and standard of software development in within time frame and resources and also modularize the application development [1,24]. Component-based development approach provide reuse the functionality of components and also integration of different components and provide the supporting system for certain functionality and achieved by the used of certain assembly of components through corresponding combination of logics and methods to identify the end-users objects and provide object based solutions for their requirements and feedback regarding the end-products behavior and functionality [19, 24].

Component-based software designing core focus area is software process designing on predefined specific objects and requirements as shown in below figure 1.11 and its major phase is component pool that manage all the desiging process and operations.

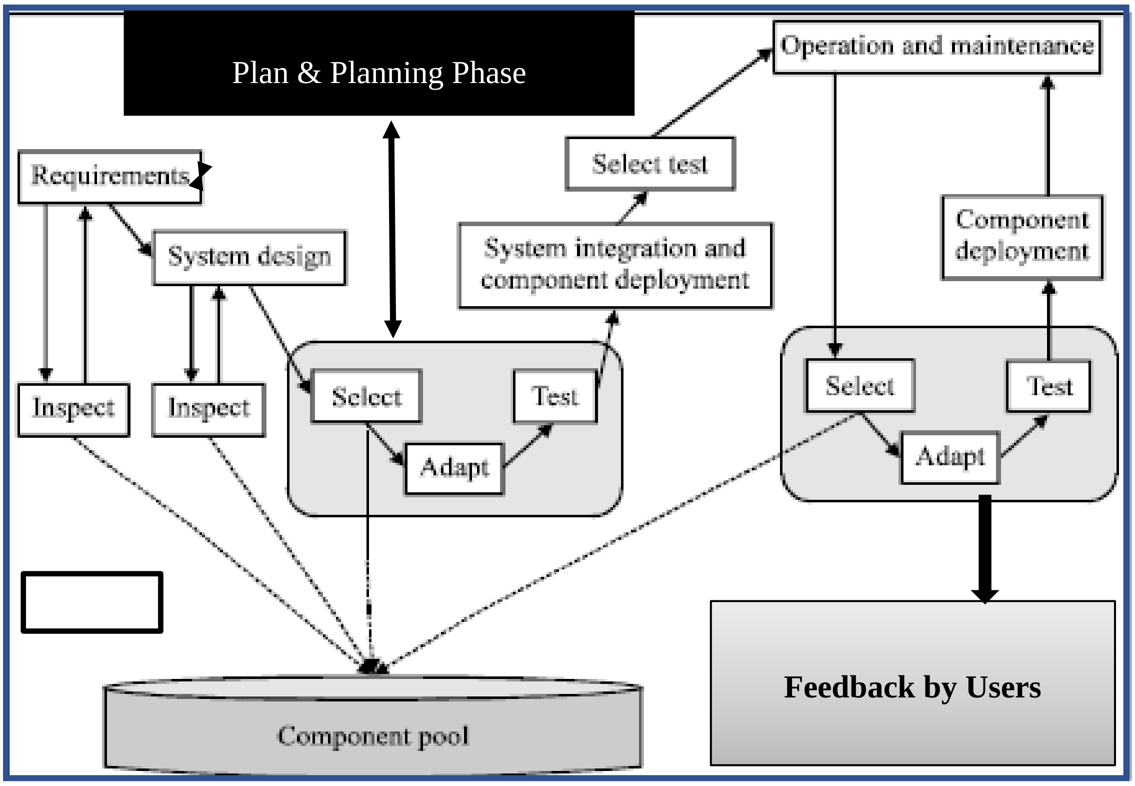


Figure 1.11. Component-based software development model

Some features of component-based development model are discussing below:

- Provide component-based framework
- Component-based development model provide supporting system based on specific objects and functionality
- Minimal the cost of software development and minimize the use of resources in software development life cycle (SDLC)
- Provide rapidly growth of software process designing and development
- Provide tools-based product designing and Modularize the tools designing
- Integration of different components and provide conceptual system
- Increase the quality and standard of software development

- Enhance the productivity and efficiency of software development
- Integration of different components within different development environment
- Provide mapping between requirement and users' feedback
- Increase the user involvement in every phase of software development
- Used different tools and methodologies for software product and application
- Reduce the dependency of different phases software development stages
- Possibility and adoptability of new methods and approaches within current software development life cycle (SDLC)
- Sometime system designing and select testing practices run simultaneously

1.1.12. Rational Unified Procress (RUP) Software Development Model:

RUP is called rational unified process software development model base on object oriented framework and model to give enhance capability to development to customize and personalized design process and activities and unified process on the basis of requirements and this rational unified process (RUP) model used the use-case driven approach, and iterative methodology software engineering to develop software naturally and reduces unexpected development costs and minimize the resources in development environment. RUP software development model gives entire software project development in a glance and it has 7 different phases of software development and management and now days this rational unified process (RUP) approach integrated with agile engineering approach extreme programming (XP) to develop software product in end-user prospective and mind set scenario and point of view [24-25]. Rational unifies process (RUP) model increase efficiency and portability by the object based driven practices and approaches and by this approach entire software development life cycle (SDLC) divided in different process environment as individual identity and then after software development team apply the different use-cases to achieved the objectives and requirements. Rational unified process (RUP) software development model capture and examine whole software in single development end and for easily understanding use the agile engineering extreme programming (XP) to deliver fast execution of the requirements and software designing.

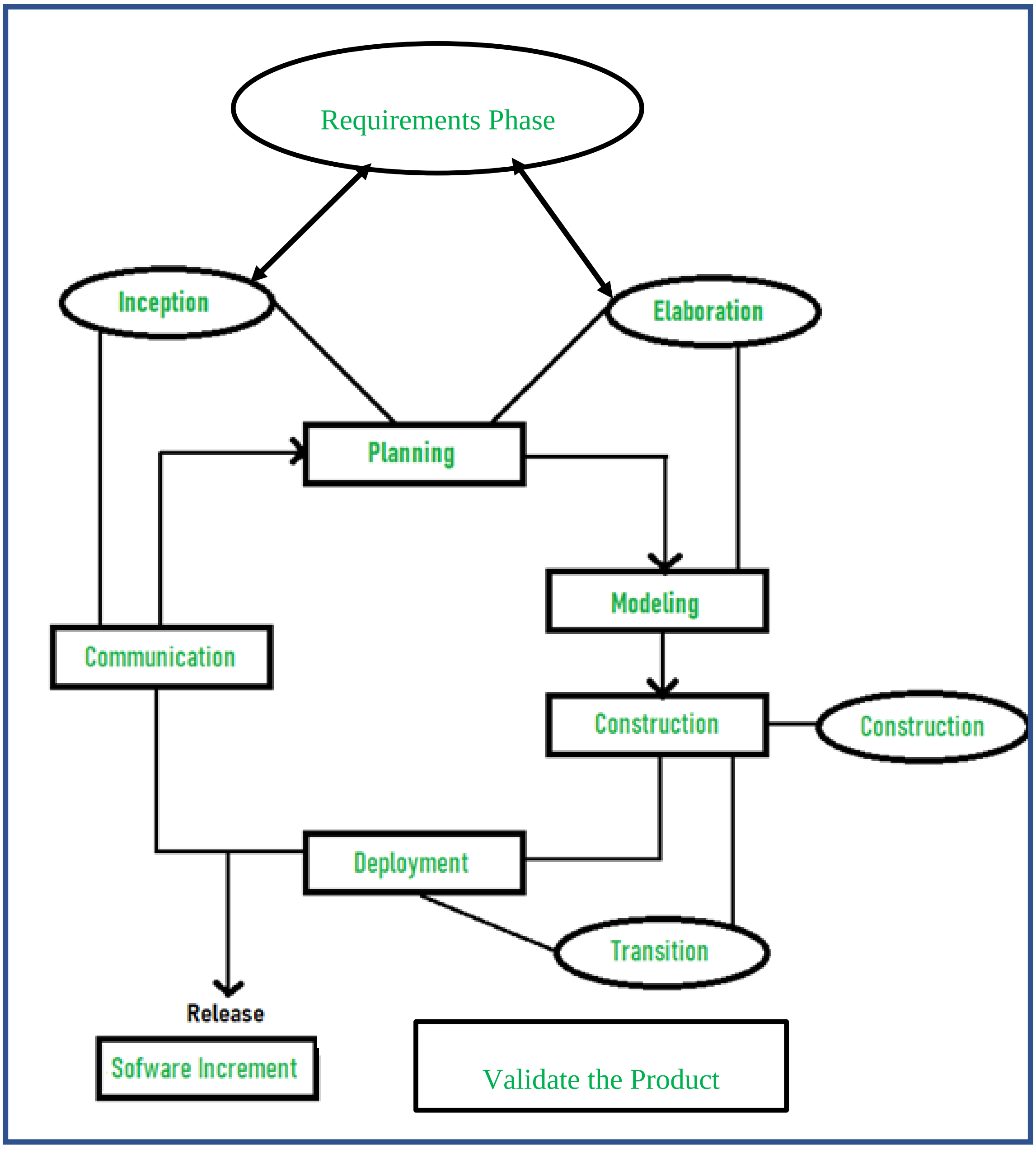


Figure 1.12. RUP software development model

Different stages of Rational unified process (RUP) software development process model and their systematically data flow approach in different phases of RUP model shown in above figure 1.12. Rational unified process (RUP) development model basically acquired the user feedback and requirements and changed in software designing methodologies and approaches accordingly the same and RUP process model increase efficiency and effectively in software development because the inception, elaboration, construction of designing process, verification and validation run parallelly in this approach.Rational unified process (RUP) designing increase the productivity according to the end-users environment and also decrease time efforts and reduce the date complexity during the process desiging and developmentation.RUP designing model enhanced the conceptual development working environment in process designing and management and also increase user oriented multi programming envrioment. Rational unified process(RUP) model work on declared data set values/attributes and undefined on demand parameters and requirements releases by end-users and stakeholders to increase the product statisfaction and quality of process/methods.

1.1.13. Rapid Application Development (RAD) Software Development Model :

Rapid application development (RAD) is software designing methodology for development software by the used of reusable process components and methods in software development life cycle (SDLC) and RAD model basically inherit the property of incremental software development approach and used business modeling, data retrieval modeling techniques, process modeling and emphasize specially on short time schedule software project in which software functionality demands in short period of the time, such as 3 to 4 months and RAD software development approach not be apply in those software project where risk priority is higher and also avoid 3’rd generation of software development because in RAD approach designing components and interfaces are changed very portable and dynamic manner in frequently time span [26]. Rapid application development (RAD) software development approach mainly utilize the process requirements specification scenario and behavior in initial phases of software development and designing to increase functional level independencies (FLI) in software development life cycle (SDLC) different phases and ensure the integrity of software quality and effectiveness according to stakeholders business interest and end-user point of view and operational level. Rapid application development (RAD) approach conceptually divided software phases in different types of tasks and subtasks for easily development and process handling and by using this approach software development and process designing industry encapsulate the different functionality on single platform and domain. Some core working environment of this approach are: Increase effectiveness by assigned and run multiple tasks on single environment, and designing of single tasks on multiple development environment on same time or in simultaneously way of designing, integrated build in conceptual level to managed the resources and time-span also.

Rapid application development (RAD) working environment and different phases are shown in below Figure 1.13. Rapid application development (RAD) approach some fundamental features are discuss below:

- Takes user requirements directly in development phase
- Elicit the requirements specifications
- Analyze and modularize requirements
- Divide modules in different team environment for process development
- Integration of all modules for system implementation phase
- Test the system for final product delivery & involved feedback from users.
- Capture and analyze the end-user responses and validate the design process
- Requirements specification scenario changes and pass to new phase of process development and forward to next level of designing team.

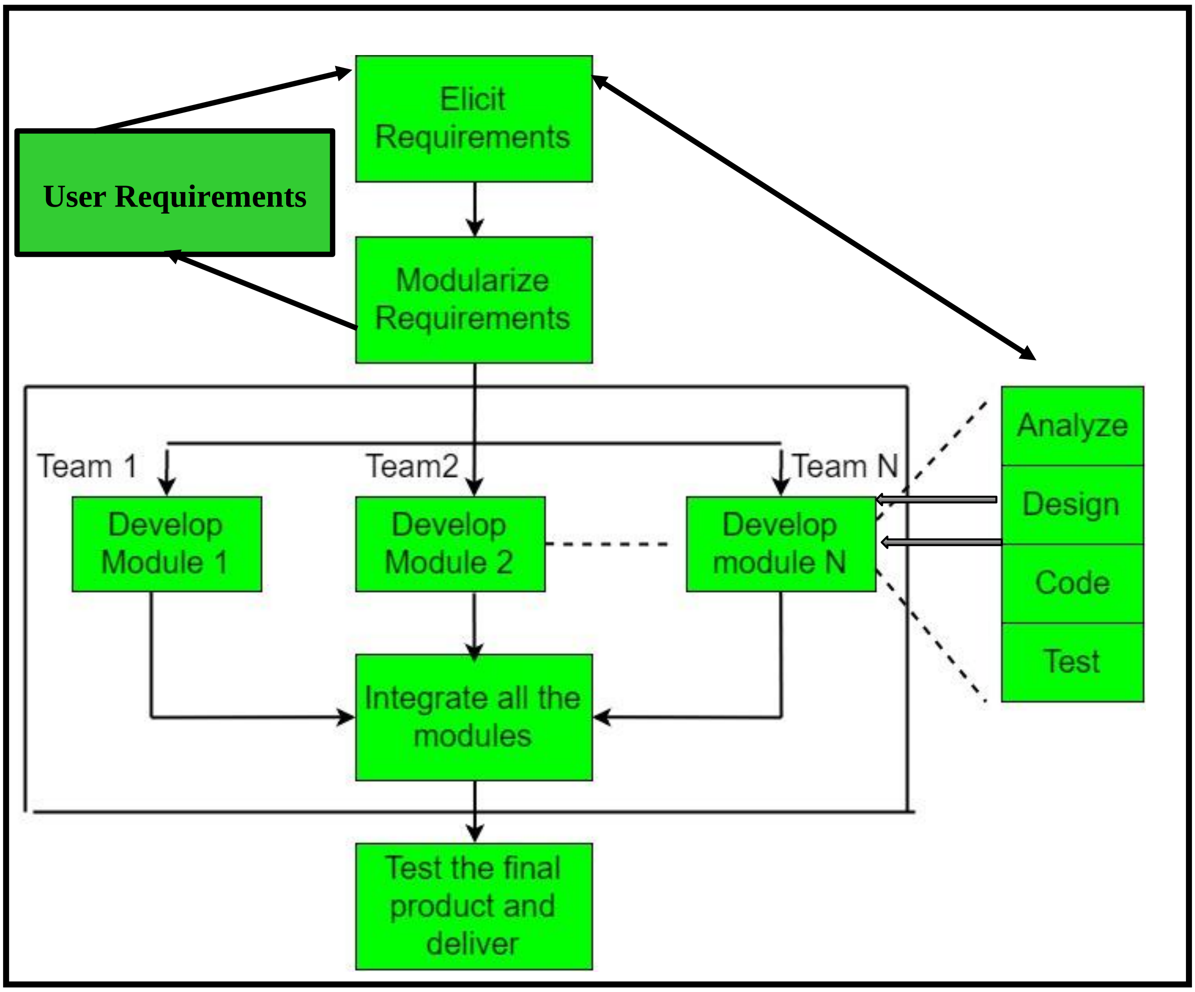


Figure 1.13. RAD software development model

1.2. AGILE ENGINEERING APPROACHES:

Agile engineering provides standards, methods, significant tools and software development framework to develop software effectively and efficiently manner and in minimum time span and in quick response manner. Agile engineering provides solutions for the end users point of view and scenario and gives internal communication between development team members and project development phases and their environment to developed standard software and in given below some of agile engineering software development methodologies listed:

- Extreme Programming (XP)
- Agile Modeling Language (AML)
- Agile Scrum Approach
- Crystal Methodologies Family
- Adaptive Software Development
- Feature-Driven Development (FDD)

Agile engineering provide light weight programming environment and agile engineering fundamental core working principal is developers relying on technical and logical excellence of software development approaches and try to produce as simple as development design on the basis of end-users needs and feedback at regular short time interval and believed to give customers greater value and quality of software [27-29] and agile engineering also increase and emphasized on internal communication, simple process designing environment, users' feedback and responses, and courage towards the development evolution and involvement [27]. Given below we discuss some features of agile engineering:

- Increase productivity according to end-user requirements
- Minimal the resources and time span
- Used innovative ideas and thought for software development
- Integration user feedback with onsite software development environment

- Increase development team scenario in respect to technical feasibility
- Increase the efficiency and effectiveness of development phases in respect to end-users' feedback.
- Involved iteration and acceptance test phase for quality assurance of software development.
- Planning can be done through spike and requirements stories phase of extreme programming (XP).

1.2.1. Overview Of Agile Xtreme Programming (XP), Agile Modeling (AM) And Agile Lean Methodology:

Extreme programming (XP) relase test scenarios, requirements stories, spike and finally release planning for iteration phae for quality fixing and then final release of process and product according to end-users requirements. Extreme programming (XP) increase the re-engineering software development mechanism to catch validite the process designing, data modularity and integrity to develop standard product and application.

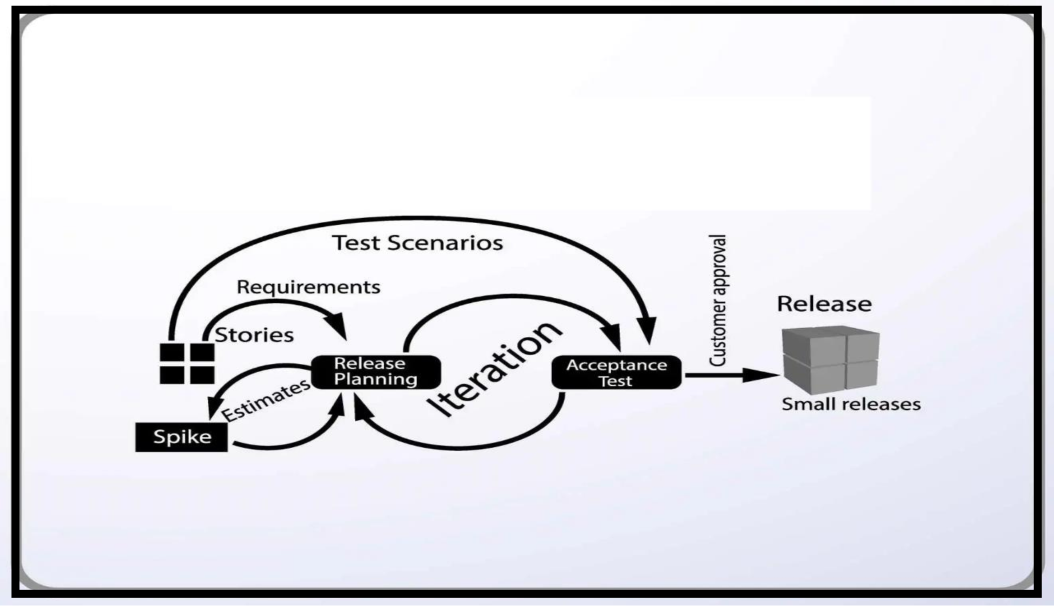


Figure 1.14. Agile engineering extreme programming (XP)

Above Figure 1.14 shwon the iteration level and test scenarios steps of agile engineering extreme programming (XP) and also showing the stories of requirements and release of customer requirements.

Different stages of agile modling Language (AML) and their project specification and increment phases of user requirments and project release, abort/retire levels during software development as shown in given below figure 1.15 and also showing the relationship between process iterate level and analysis phase.

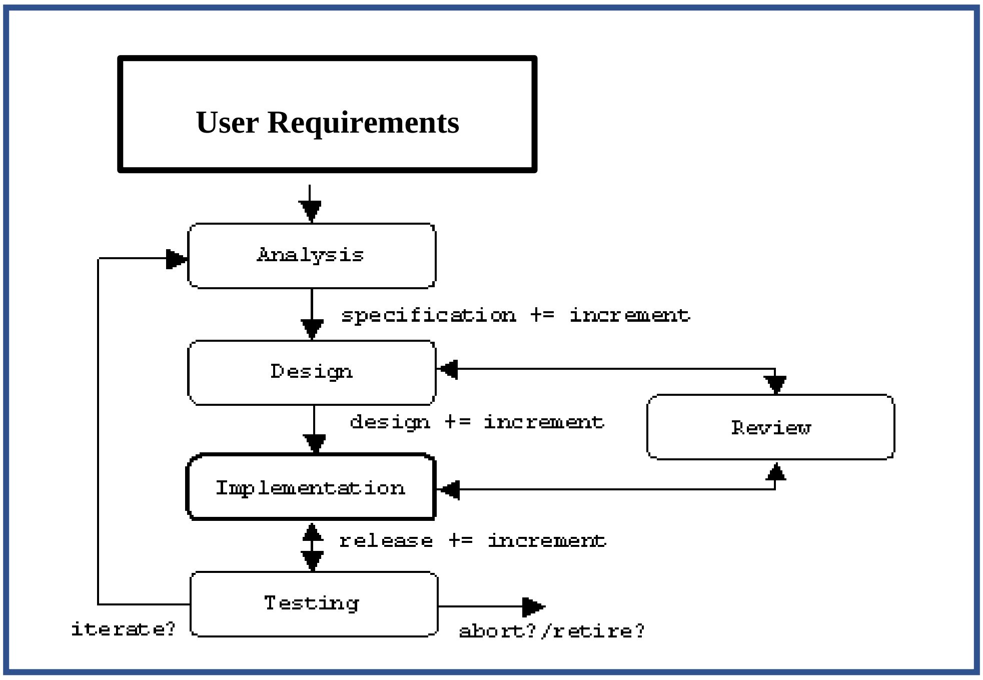


Figure 1.15. Agile modelling (AM) software development approach

Lean software development approach is agile engineering concept that is mainly used to synchronous and optimize the software development and process designing. This development technique released by Toyota Production System and developed by Taiichi Ohno in the 40's in the Japanese production industry context and worked on minimum viable product (MVP) methodology and core focus of this technology is streamline production and minimal waste [36, 37]. Some of the core working principles of lean software development methodology are defined as below:

- Value
- Value stream
- Flow
- Pull & Perfection

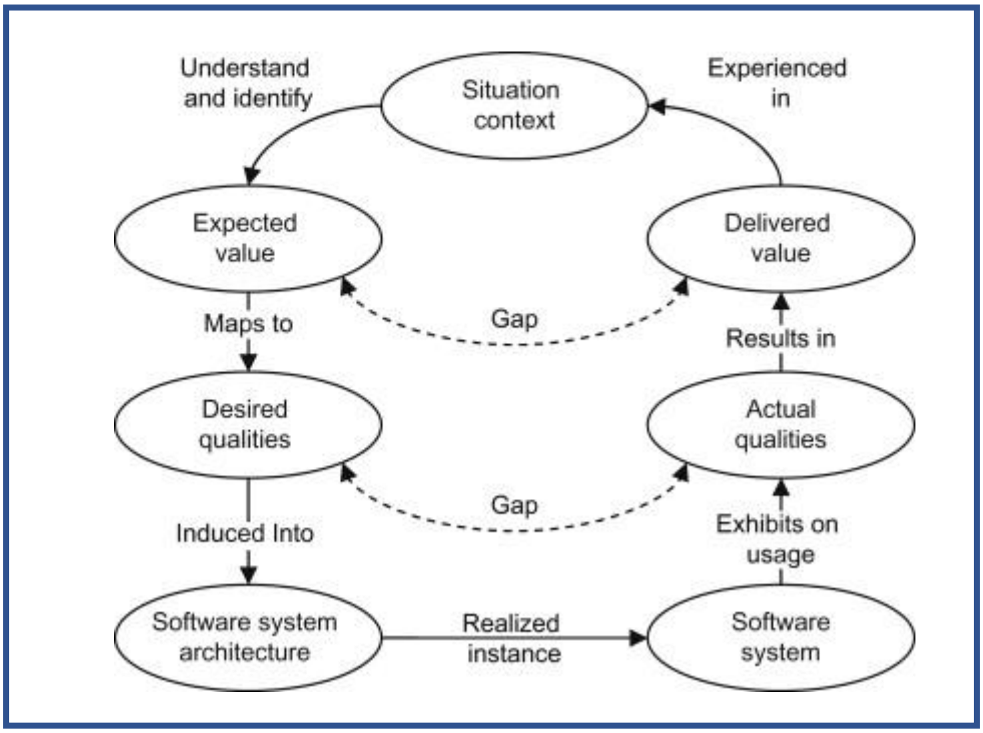


Figure 1.16. Lean software development working environment [36-38]

Lean software development working environment start with understand and indentificatin phase of user requirements and findout the gaps between process designing and maps the outcomes in respect of desired qualities and attributes as shown in above figure 1.16 and also give view on the streamline working environment of lean software development different phases and process level stages

1.3. DESIGN THINKING (DT) APPROACH CORE ATRIBUTES:

Design thinking (DT) is approach to provide solution for issue finding, forecasting of high degree of awareness and understanding of end-users needs and demands and given new methodologies towards the "iterative process designing and development and focus on minimum viable product (MVP) in respect of human centred software development". Design thinking (DT) giving a new framework for service designing (SD) and provides solutions for novel challenges in end-users' expected software designing and always concern on highly endeavor innovative working environments within the development team [30-32]. Design thinking (DT) generally provides graphical representation of end-users demand by the process designing and development and inoloved their mindset and feedback in intial stages of software desiging and developmentation. Design thinking (DT) conceptually level provides road map for human centred software development and statisfaction level of real users.

Desging thinking (DT) approach work on five different core attributes as listed below:

- Empathize
- Define
- Ideate
- Prototype
- Test

Figure 1.17 givne below shown the design thinking (DT) core attributes and also showing the their data flow within frame of design thinking (DT) working domain.These five core attributes of design thinking (DT) play significant role for human centred software development and designing. These attributes of design thinking (DT) provides effective and acceptable solution on end – users operational operational point of view and for this design thinking (DT) use requirement engineering (RE) as a platform growth and awkward process designing environment for human cented software development (HCSD).

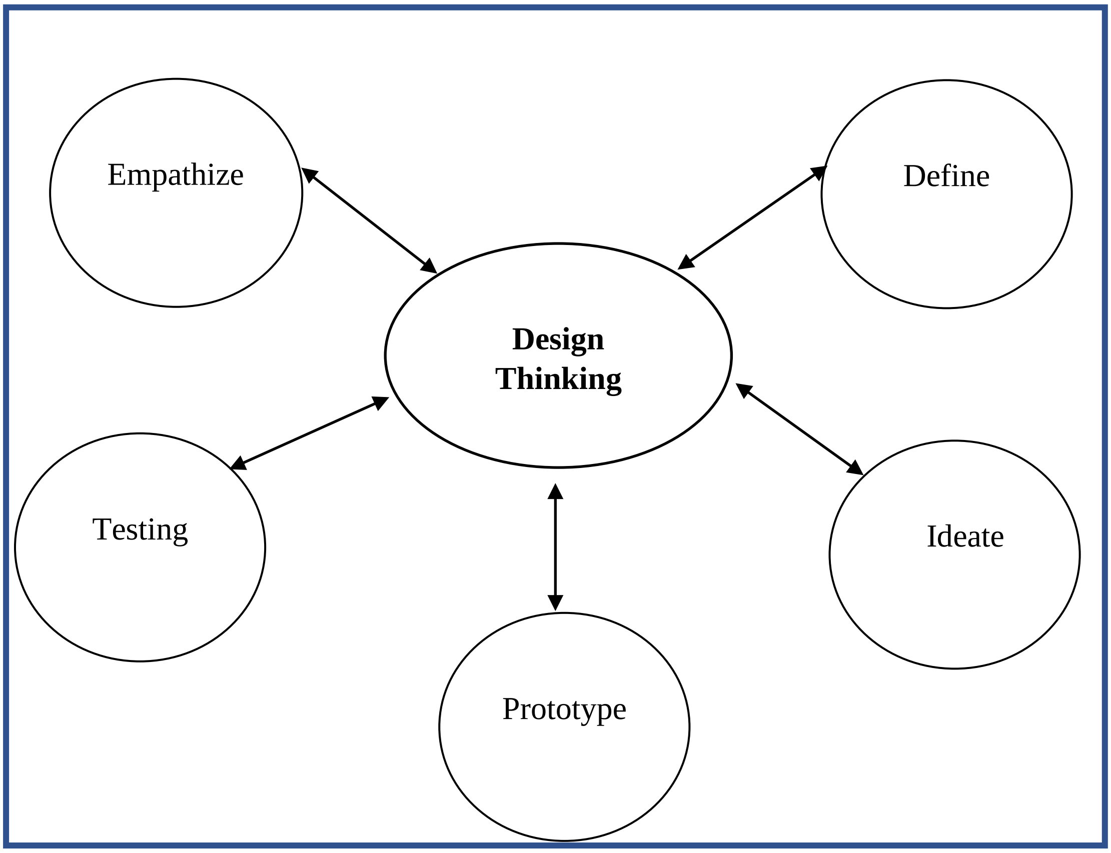


Figure 1.17. Design thinking (DT) approach core attributes

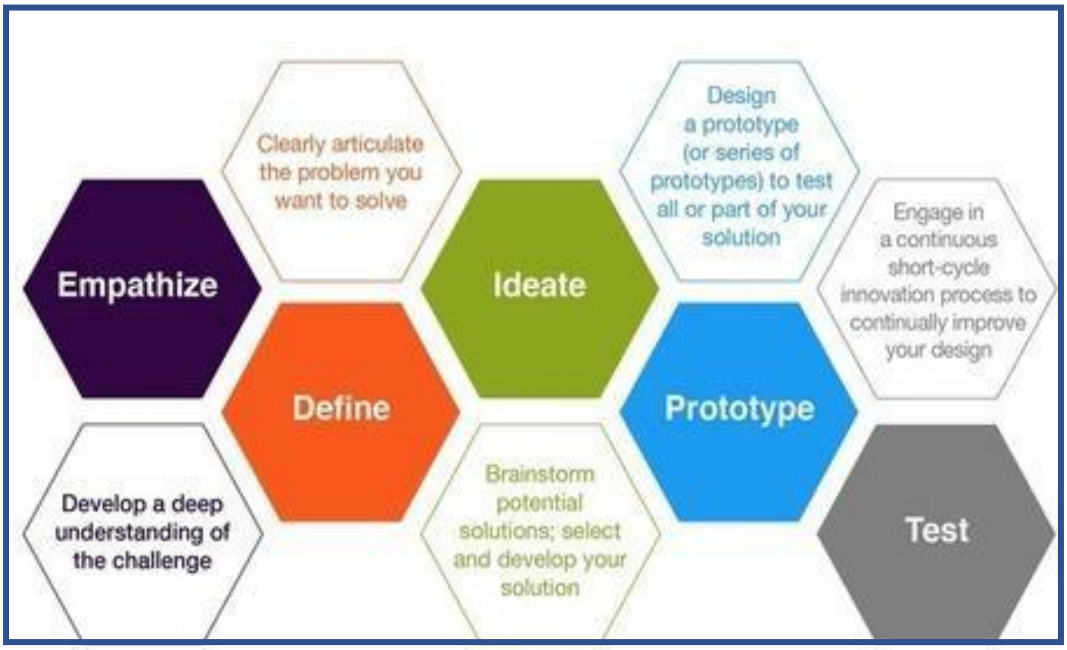

Clearly articulate the problem you want to solve
Design a prototype (or series of prototypes) to test all or part of your solution
Engage in a continuous short-cycle innovation process to continually improve your design
Empathize
Ideate
Define
Prototype
Develop a deep understanding of the challenge
Brainstorm potential solutions; select and develop your solution
Test

Figure 1.18. Design thinking (DT) features

Above figure 1.18 gives view on different features of design thinking (DT) and their intejoin of designing properties like deep understanding of requirements, design prototype, short-cycle processing,clearly articulate for problem phase.

1.3.1. Role Of Requirement Engineering (RE) In Innovative Envrionment Of Design Thinking (DT) :

Success and quality of software product development depends how well it's fulfilled the end-users operational and behavior needs and its environment too. Requirement engineering (RE) provides the platform and solutions by which we understand and validate the users' needs and their expectation from the software development process and engineering and alos make innovative environment for design thinking (DT). Requirement engineering (RE) is an approach of software development which provides ethnography and synthesis problem solutions towards the goal and objective specific [34]. Requirement engineering software development approach mainly emphasize on product requirement specification and provides optimize solution if requirement will be change in designing phase from the stakeholder and operational users environment. Requirements engineering (RE) make triangular relationship between user satisfaction, users needs and validation phases of requirements to produce optimize solutions for product delivery and designing. Requirement engineering (RE) core attributes and their relationship in respect of customers feedback and user statisfactions always play vital role in verifications of software designing and development phases as shown in below figure 1.19 and also gives view on designing data flow order and also showing the relationship ordes in different edges of requirement engineering (RE) and user satisfaction factors always on top priortity in this approach.

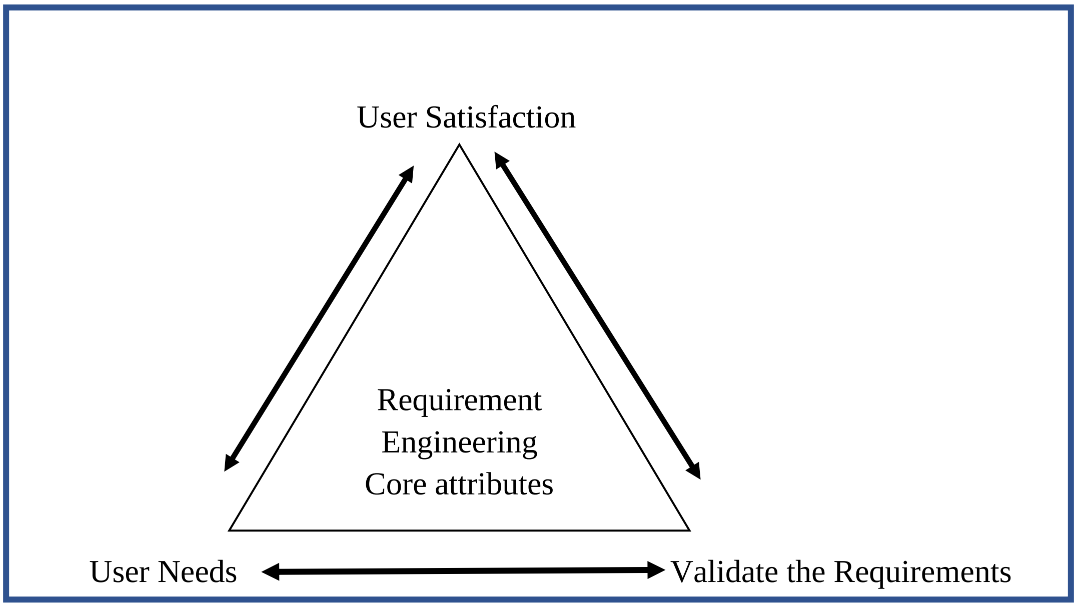


Figure: 1.19. Requirement engineering (RE) software development approach

1.3.2. Requirement Engineering (RE) Attributes:

- Depth understanding of end-user’s requirements
- Requirement specification for problem and requirements
- Empathize of end-solutions
- Procedural layout of problem specification
- Verification and validation of requirement prototype
- Decided implementation level according end-users
- Work on dynamic requirement phases
- Provides interaction channel between requirements and process designing
- Try to involve in middle of software development phases
- Reduce the process designing complexity, save time and resources
- Requirement engineering work on empathize working environment
- This approach produce real system behavior environment on end-user needs
- In this approach user acceptance index factor is between 4-5 (out of 5)

Requirement engineering (RE) produce sophisticatd solution and also increase user acceptance level. Givne below table 1.1 shown the relationship between SDLC phase, activites carried out during designing process and their user acceptance level range.

Table 1.1. Requirement engineering (RE) activities and user acceptance range

PHASE NO	ACTIVITIES /OBJECTIVES	User Acceptance Level Range (1-5)
I	One Line Requirement Study and Empathize on Future Work Layout.	1-2
II	Understanding, Specification, and Produce Real System Behavior for Development.	2-3
III	Sketch, Decide Prototype Model, and Validate According to End-User Needs.	4-5
IV	Synthesis, Ethnography, and User Oriented Prototype Model.	3-5
V	Define Operations Approach and Apply Solution for Validation Purpose of Client/Stakeholder Requirements/Needs.	2-4

1.4. HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) APPROACH:

Human centred software development methodology not only concern about the changeable world of requirements due to proactiveness involvement of end-users and stakeholders during the software development life cycle (SDLC). Human centred software development (HCSD) approach also imperial work on develop product quality assurance by taking frequent and constant level of feedback support system within different development phases for this human centred software development (HCSD) and increase process creativity,innovation ideas, and real operational data vision from the user operational standards and mindset. Human centred software development (HCSD) approach adopt dynamic human-centric software development mechanism and approaches [33] and increase sytem satisfaction and specified requirements in context of enduser operational environment as shown in given below figure 1.20. Human centred software development (HCSD) provide real operational behavior environment and taking feedback directly in requirement specification phase to provide dynamic solution.

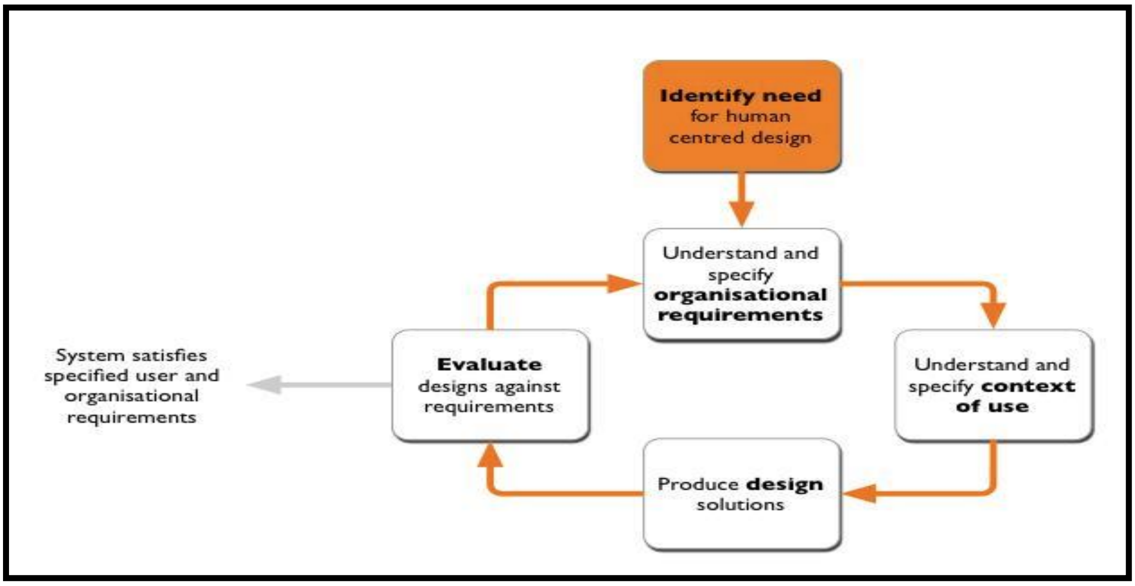

Identify need for human centred design
Understand and specify organisational requirements
Understand and specify context of use
Produce design solutions
Evaluate designs against requirements
System satisfies specified user and organisational requirements

Figure :1.20. Human centred software development (HCSD) working envrionment

1.4.1. Features of Human Centred Software Development (HCSD) [34]:

- Creating software by human-centric endeavor approach
- In this approach build software directly used by end-users
- Human involvement increases the efficiency in SDLC
- Psychological heuristics apply in software development
- Central task management system approach
- Individual interactions and functionality methodology
- Abstracting the most important functionality behind the screen
- Reducing cognitive features
- Involved human centred feedback and communication in SDLC
- Human centred design (HCD) and Use rational unified process approach
- Increase usability engineering (UE) in software development
- Project finished in decided project scheduling and cost estimation
- Used agility engineering approaches to produced standard software
- Integrate the design thinking (DT) technique to developed human centred software development (HCSD)
- Internal communication system built up within software development for user acceptance software

1.4.2. Traditional Software Development Vs Human Centred Software Development (HCSD) Approaches:

Traditional software development generally working on components driven and other end human-centred software development (HCSD) approaches work on solution oriented and user-driven and given below table 1.2 shown the comparative studies between traditional and human-centred software development (HCSD) methodologies.

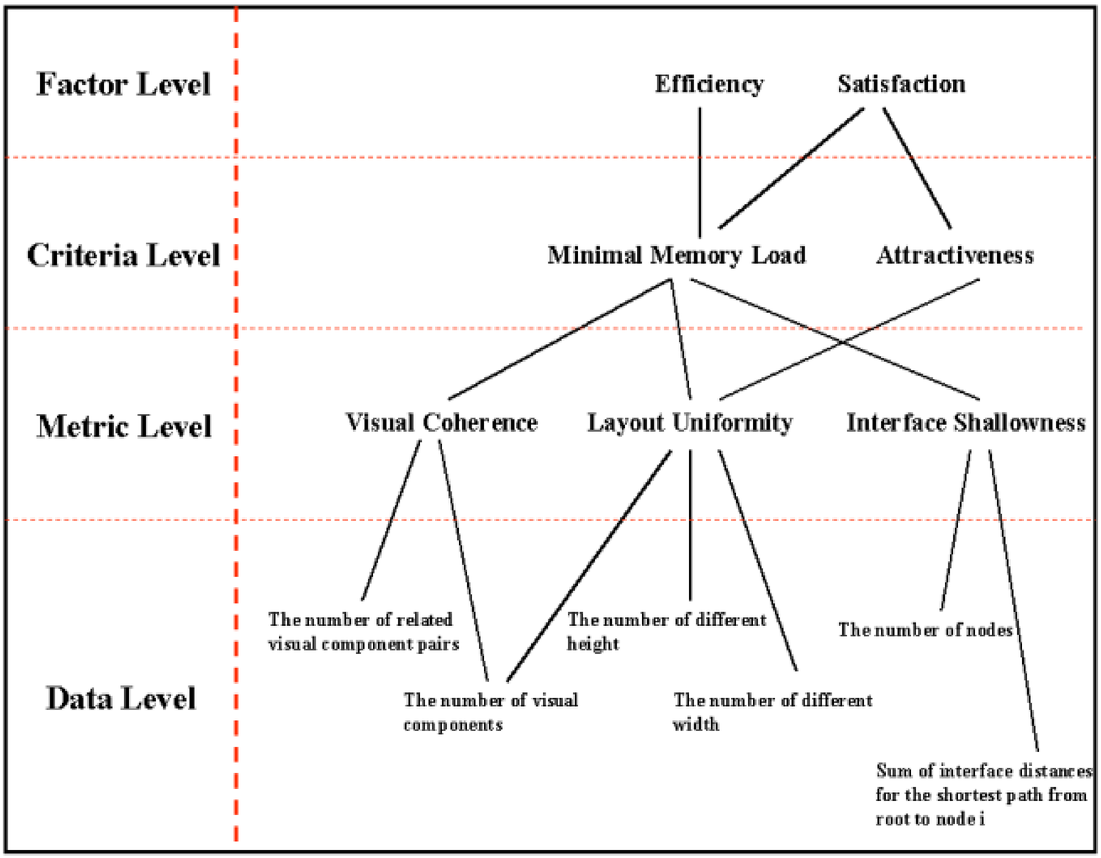
Table 1.2. Comparation between traditional & HCSD approaches

S. No	Traditional Software Development	Human-Centred Software Development
1.	Technology/developer-driven	User-driven
2.	Component focus	Solution focus
3.	Individual contribution	Multidisciplinary teamwork
4.	Quality measured by defects	Quality defined by user satisfaction
5.	Strength on internal structure	Strength on external Structure
6.	Functional requirements	Understanding of context of user

1.4.3. Architectural Human Centred Software Development (HCSD) Factors:

In software development life cycle (SDLC) architectural framework and user level satisfactions divided in four level user interaction to increase efficiency, satisfactions, attractiveness, and also enhance the coherence factors and interfaces level in software development.Architecture framework generally divided in 4 phase: factor levels, criteria level, metric level, data level of process development and designing and also count the minimal memory load and attractiveness and sum of interfaces distances and width values. Architecture framework mainly focus on customer statisfaction, methods efficiency, visual coherence of data values and focus on shortes path from the requirement analysis to final data model. Architecture development framework provide visual data conceptual layout and also provides number of different width of end-users data values.

Architecture framework factors core focus areas are efficiency and satisfaction and fix the data integrity on different width values as depicated in figure 1.21 and come across the four level of designing and also work on memory load and uniformity of designing phase.



Factor Level
Efficiency
Satisfaction
Criteria Level
Minimal Memory Load
Attractiveness
Metric Level
Visual Coherence
Layout Uniformity
Interface Shallowness
Data Level
The number of related visual component pairs
The number of visual components
The number of different height
The number of different width
The number of nodes
Sum of interface distances for the shortest path from root to node i

Figure: 1.21. Architectural HCSD framework and user satisfactions factors

1.4.4. Process Development And Testing Predication In HCSD:

In software development life cycle (SDLC) environment, the quality and acceptability of software application/product is depends on process designing different phases and their software testing prediction parameters and assurance standards. Software quality effectiveness and efficiency increase and gain by different aspects of software methods, process specification and quality in use through the verification, validation and methods of statisfactions and testing predications mechanism in software developments and process designing. Methods and process strategies make integration between requirement specification and testing predication to provide quality solution.This testing predications strategies develop quality in different phases of human centred software development

(HCSD), through primary and secondary artificats specifications in methods and process levels of software development and designing.

Methods and process specification statges divided software development life cycle (SDLC) in primary artifacts and secondary artifacts for specification document and user manual and compute prototype model for final system production stages as shown in given below figure 1.22 and better process management and software development.This methods and process specifications stages gives artifacts for human centred software development (HCSD) and ensure high fidelity protoype designing environment.

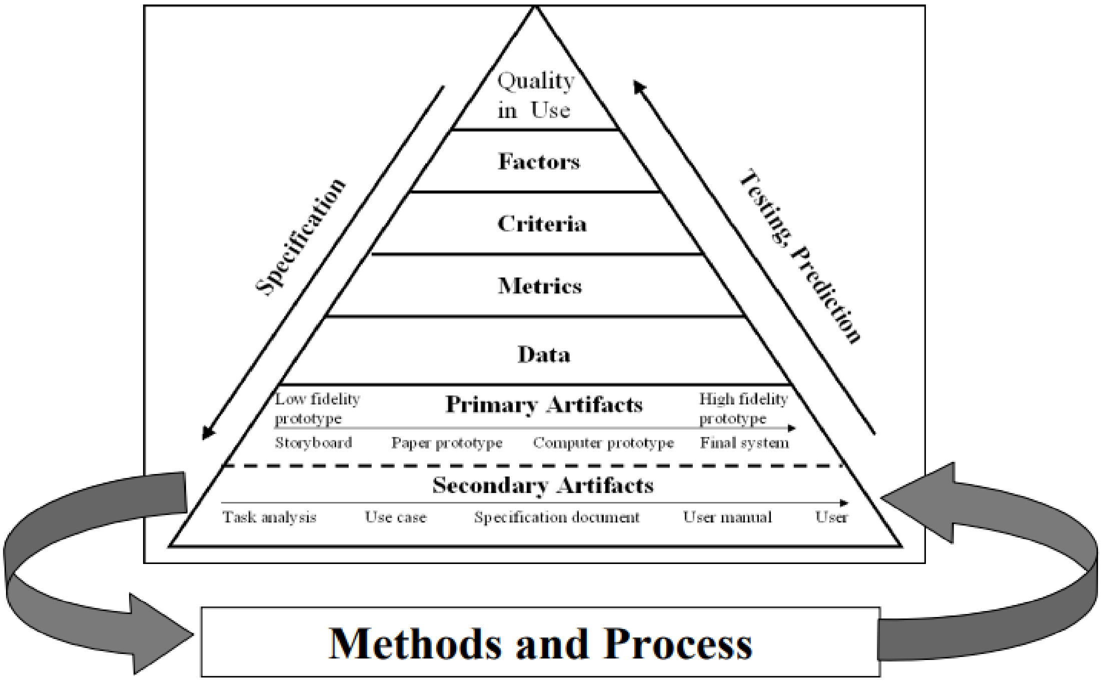


Figure: 1.22. Methods and process development specification stages

1.5. MOTIVATION:

All the traditional and modern software development models and approaches have some advantages on their end but not be give complete solution on the basis on real user’s operational environment and also full-fill the customer’s requirement under one roof of development, such as traditional approach provide better solution but if requirement is fix and static manner. Agile engineering software development approaches generally develop software in designing sprint form to involved user feedback and responses as earlier stages of software designing and development. So, these challenges become vision of this research to design and develop software process management metamodel using agile-human centric and design thinking (DT) approaches to give completeness and validity in software development and satisfy end-users requirements and operational behavior too.

1.6. OBJECTIVES:

The objectives of this research work are:

- To understand, investigate the strength and weakness of existing software development approaches.
- To design and develop a process management metamodel based on agile-human centric & design thinking (DT) approaches to increase end-user satisfaction and improving productivity.
- To evaluate and compare the developed human centred software development model (HCSD) with existing models.

1.7. CONTRIBUTIONS OF THE THESIS:

The contributions of the thesis are listed as follows:

a). Discuss the different software development methodologies and their working environment and features and to analyze the software development challenges and also discuss and analyze the merits and demerits of traditional and modern software designing and development technologies according to end user-point of view also and working environment and business stakeholder objectives and interest.

b). Understand and comparative studies of agile human-centric software development approaches and design thinking (DT) approaches to create prototype model on the basis of end-user feedback and responses and involved them in design phase of requirements and takes their suggestion to develop prototype model according to end-user's requirements and business stake holders' objectives for effectively.

c). Human centred software development approaches and their working algorithm and framework towards human centric software development and quality indexing.

d). Give the detailed view and architectural framework for "design and develop conceptual integrated process metamodel using agile-human centric and design thinking (DT) approaches" and their working features and environment.

e). Result and Discussion phase of design and develop process management integrated metamodel based of agile-human centric and design thinking (DT) through user acceptance index level and also in respect of cost and time index factors.

f). Give the summary and conclusion of reseach work and also discuss the future scope of design and develop conceptual integrated process metamodel using agile-human centric and design thinking (DT) approaches.

1.8. ORGANISATION OF THESIS:

This thesis is organized in the five chapter.

Chapter 2 in this chapter completes Literature review and related work of this research work and also takes idea and input for design a conceptual metamodel.

Chapter 3 in this chapter we discuss the Agile-human centric and Design Thinking (DT) software development approaches and their significant role in Human Centred Software Development.

Chapter 4 in this chapter we discuss the framework and working environment of Human Centred Software Development Conceptual (HCSD) Meta Model and also discuss the different features of designed and developed human centred software development (HCSD) model.

Chapter 5 in this chapter completes Implementation, result and discussion.

Chapter 6 in this chapter conclusion of the complete thesis work & future scope.

Chapter 7 in this chapter gives the references.

1.9. CONCLUSION :

Traditional software development (TSD) methodologies generally provides statics empirical solutions for software developments and in these types approaches of sometimes software development team members not carry out realistics requirements of end-users end. Agile-engineering software developments approaches always focus on quality and dynamic designing edges of software development to save time and complexities of data integrity in desginging and in other design thinking (DT) provides graphical represents of software development and inovled users end innovations and methods to gives standards portable solutions. Human centred software development (HCSD) approach emphasized on users feedback and responses involvements process in every phases of software development to release sotware product according to end-users and stakeholders statisfactions levels.In the last of this chapter discuss the background of this research work and also gives objectives of research work and discuss the organizations structure of the thesis.

CHAPTER - 2

LITERATURE REVIEW

LITERATURE REVIEW

This literature review based on quite dependent on conceptual, informative and after this comprehensive study and analysis some reporting facts, information and evidence come out regarding the different traditional software development approaches and modern and human centred software development methodology and their significant impact in software development life cycle (SDLC).

2.1. PAPER SELECTION FLOW AND SYSTEMATIC LITERATURE REVIEW (SLR) :

From this systematic literature review (SLR) we provide the concrete knowledgeable view and data that come across globally in this research, literature review are paper selection process in this research total of 1076 papers appeared from our initial literature searches in the specified database then relevant papers selected form the reading of title and abstract: 280,relevant papers selected after removing duplicate and applying inclusion and execution criteria: 220 and in last final set of paper after applying quality criteria is: 167 papers and given below figure 2.1 shown the flow of the paper selection criteria and systematic literature review (SLR) process of research.

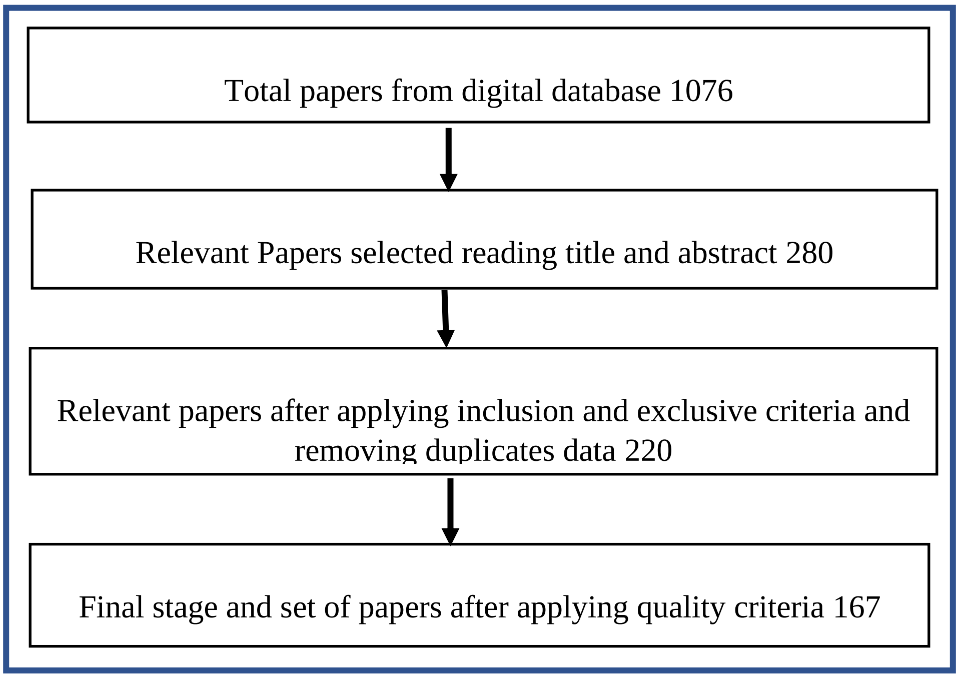

Total papers from digital database 1076
Relevant Papers selected reading title and abstract 280
Relevant papers after applying inclusion and exclusive criteria and removing duplicates data 220
Final stage and set of papers after applying quality criteria 167

Figure 2.1. Flow of Paper Selection Process in Literature Review

1. **According to Minelli [39],** the task of process and program understanding is gain more 50% of all the user acceptance maintenance activity. The core reason for this is the not clearly declare and defined the abstractions methods on the application, in respect of thousand lines of code (LOC) come across in hundreds of data files, it will become more difficult task to understanding of this statement of codes.

2. **Kaisa Savolainen [40],** for any kind software project, we analysis and reviewed methods/tools that fulfill and allow building abstraction of higher-level software designing, such as architectural level views designing, we examined tools that get commodity views designing such like UML (Unified Modeling Language) class and package structure diagram from object-based Java source.

3. **Lano,Rahimi,Troya, and Hessan [41],** have proposed and emphasis on agile model driven engineering impact value was one of the crucial and significant leading designing components and forces in respect the adoption of most agile engineering software designing and development approaches. This objective of study and analysis result point out the importance of lean software development approach liking with kanban and elimination of unused data and process variable with continuous learning goal and cycles and come true the fast iterations and product delivery.

4. **In this Andriole[42],** has given view in context of business environment now shift to digital software development transformation era, in which disruptive end-users business solutions for process and development design models are mandatory to give competitiveness, especially for those business world environment that are not support and willing to give effective and significant monitoring and controlling methodology to their designing processes to large group of software vendors groups and end-users environment and also reduce technical complexity and also reduce inflexibility of large size software and process designing too.

2.2. KANBAN THEORY :

5. **According to Anderson [43],** the lean methodology uses a number of technical tools to managed and support software development management and its operational level. In those tools one tool name is Kanban worked as capital K technology in software development and given by Anderson and the K methodology mainly focus on

visualize the process flow of a production designing and operational environment and use queue theory to control, monitoring and improve the value system stream by given special attention at the designing area of software development and production integrated flow within different phases of software designing. In software engineering David Anderson, is first person who introduced and used **“Kanban theory in software development”** with the association of Microsoft and he giving five working principle for it:

- Visualize workflow
- Deploy limit in process designing
- Control and manage the development
- Design explicit policies for software designing
- Focus on improvement area in software development

6. **According to Corona and Pani [44],** kanban methodology usually used wall of implemented board matrix mechanism in their columns showing and implementing the different development methods and process stages. Cards points out and used to showing piece of unit of work or tasks, which come across by the chart columns and in here column generally representing the stages of work specification, process development, test and implementation and each column also emphasize on the daily issues in work flow environment, crucial requirements and needs and finding the improvement area in resources opportunities and optimization solution towards the objective materialized within process designing.
7. **Santos and Travassos [45],** have proposed study in which they discussed factors and values those are benefits and challenges of using kanban approach in software development for regarding this the result of primary studies on kanban technique aggregated with “**structure synthesis method(SSM)”** and in this SSM allows the integration of quantitative and qualitative and in this SSM point out briefly on core contextual aspects and knowledge based impact trends(e.g. positive information or negative information), as well decided the certainty designing estimation about the process development and scheduling of estimation and it also provide balance positive phenomena in respect of resource optimization finding level and their declaration for project development.
8. **According to Matkovic & Tumbas [46],** software development life cycle(SDLC) methodology has been given a model of development for plan, requirement analysis,

designing, quality control and testing, implementation and post implementation maintenance and for this software development methodology used technical description of process designing and development approach as a sequential or iterative manner.

9. **In this Wright [47],** has been proposed that software development approach provided direct point to point to manage the software project complexity and also work and implications for user acceptance usability, adaptability, portability, maintainability, reliability, and quality acceptability in reference to developed software and in opposite if wrong approach adopted for software developmentation then user dissatisfaction increase, lack in quality validation, administration overheads, and in brief direct significant relationship between adoption of standard and correct software development methodology and customer and vendor satisfaction in software development life cycle (SDLC).
10. **According to Permilla, Eli, Richard [48],** in the Covid-19, agile software development approaches changed software development industries prospective and addresses on some pitfall of traditional software development techniques like heavy project documentation, not matched decided productivity, user acceptance, reliability and lack in designing simplicity methodology and lack of risk analysis and also not produce standard and quality software product. Inspite significant uses of agile engineering in software development in last 10 years, agile software development methodologies have also point out many demerits and some of them are dependent on run-time techniques in comparison to more documented knowledge of data processing, lack of adequate man power if project size is increased and highly skillful team required to handle this type of project and high degree of operational implementation limitation always come forward in software project designing.
11. **In this David Itzik and Gelbard Roy [49],** discussed the outcome of software designing and project development based on agile engineering approach mainly dependent and concern on organizational factors like end-user commitment, decision time to designing and implementation, project team size and environment composition, customers personally involvement in software development life cycle (SDLC).
12. **According to Dyba & Dingsoyr [50],** have proposed that agility identification and analysis depend on various methods and approaches and drawing to give model which is based normative thinking about development strategies to deciding and choices and

their relationship and practicing managers to adopt agile methodology and also given knowledge based body of development, where empirical knowledge have been fixed in size of scope.

13. **Sarah Laoyan [51],** have discussed in there research that agile engineering methodology depend on incremental and iterative approach of software development and same time mid of 19th century most of software development industries worked on the practices of separating modules design and development, testing and validation of process development and then implementation of designed work and verified the end-user's expectation from the software development.
14. **Nerur & Balijepally [52],** have proposed the study in which they discussed that implementation phase of software development was decided by different generations of process designing of codes and also discussed the functional sub system specification, and there were interfaced level interfaced check points for work verification and validation at the last level of product scenario and implementation behavior of software development and for this software development started to taken advantage of evolutionary software project management work methodology divided in iterative and incremental level of product designing and to gain this objectives design system approach is break the large complex system development work in small units and process and achieved the given the prototype and goal expectation within sufficient level of provision on completed work done and environment to achieving the committed goals of customer and stakeholders from the software development life cycle (SDLC) and in brief agile engineering provide conceptual benefit and foundations in other streams of software development architecture, social technical systems, soft computing system methodology, enhance congruence and transitional industries. Agile engineering software development techniques main attributes are greater autonomy, designing techniques, conceptual and adaptive knowledge of requirements, understand the behaviors of theoretical backend and then go for problem solving framework and architecture for pre-decided goal and objectives.
15. **According to Strode [53],** agile engineering are many methodologies, which involved in various principles and processes and different to each other and very few technology work together for commonly development growth and work but all the differences, all of agile engineering approaches significant focus on business and stakeholder problem definition and their quality solutions in the minimal time-frame, and satisfy users view and always concern and gives priority on their dynamically

changing in requirements and work process on low key index and communication is always on high level of index and developed quality oriented highly committed and motivated development team.

16. **According to Abrahamsson [54],** the differences among agile software development engineering methodology mainly depend on their uses and purpose, solution they provide, end-users needs and requirements, and some of them emphasize on techniques and practices used in software development and others concern about management and stakeholder business aspects and agile engineering development approach has significant extent level of coverage for the stages of software development and also focus on team scenario and their composition for increased the productivity and respective techniques used to enhance the end-users operational environment feasibility and validation.
17. **Qumer and Henderson-Sellers [55],** have given a framework that comparing agile engineering methodologies depend on their agility features and characteristics and this architecture basically based on four conceptual dimensional: scope of requirement, feasibility features, agility features values, and processes verification and validation and in this study based on six agile technologies and in last framework come on point of agility index and which is used to finalized agile approach to accomplish the software development within decided requirement and resources and it is used as a guide and also provide the choices based methodology in software development life cycle (SDLC).
18. **According to Martin [56],** given view on that customer play vital role in software development life cycle (SDLC) through agile engineering and given key area of responsibilities to drive the project, continuous relationship and feedback with users and business partner to develop and provide retrospection system to test the interface deliverable and with its compliance and sustainable software development and quality standard. In this study authors also, focus the role of developers to drive the project integrity and effectiveness and decided the labels such as: technical driven knowledge spectrum, negotiator, client areas, and specialized designer of every phases of software development at every stages of software development life cycle (SDLC).This study summarize customers productivity and effectiveness in three areas of agile engineering: (i) actual customer environment and interface, (ii) create team-ness perspectives to build up standard software and (iii) energized process working

level and increased working effectively and productively environment to achieved decided goal and objectives.

19. **Siakas and Siakas [57],** provide strong and closed relationship framework between organizational environment and culture with agile methodologies and point out that how agile engineering is differ forms distinct development approaches. They also given conceptual model base on organizational environment as clan, democratic process stack, inherit and well systematical agile engineering approaches those suited in more democratic working environment.

2.3. DYNAMIC VS INTERNAL SOFTWARE DEVELOPMENT :

20. **Iivari and Huisman [58],** have proposed a conceptual competing view value model, a two edges model, core focus on value of processing and provide knowledge that values are main constituents of working environment and culture and model provide deep knowledge of aspects-dynamic Vs static and internal Vs outside environment of software development and also provide mapping between group working culture depend on human resource relationship and portability, development environment belong to future scenario ,and rest of values depend on rational behavior, goal specific, routine and rule regulations to capture the achievements.
21. **Iivari and Iivari [59],** have given view in their studied that value based model in context of agile engineering development environment and also pointed out in research that agile methodologies and hierarchical working culture are incompatible to each other and agile methodologies implementation is quite possible and changed working environment in hierarchical organizational culture is possible through the integration of complementary attributes of different techniques of agile engineering methods to get the specific development requirements and goals but this view of working might be increase the heaviness of implementation and due to this agility of software development sometime will be break and not capture the standard and quality for this reason.
22. **According to Chow [60],** some factors that can give direction and guidance in software development through agile engineering methodologies and also their studies they identified some attributes selection and their impact on project management in organization level and specific requirements oriented and for this purpose authors used six keys element selection process to decide the goal specific agile methodology

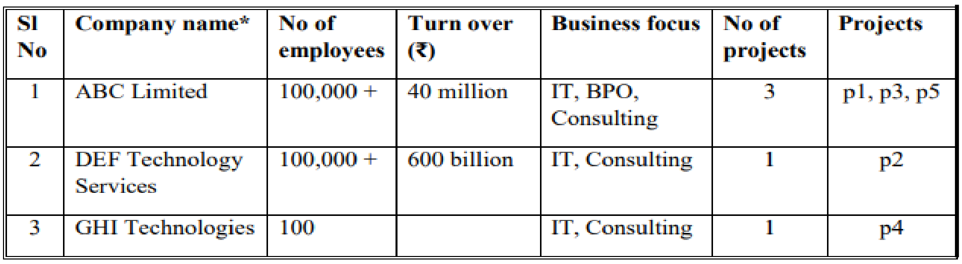
and also cover delivery dimension of product and project through re-engineering concept, strong functional testing approach, effective and competent members in development team, adaptive process management statement style, team management, communication level within team management, customers and development team feedback with open authority to each members designing and development team of software project.

23. **Aik _Ling Tan [61],** has highlight aspects of organizational environmental level agility namely, rapid changes in response level, innovation, initiative and keys of learning that are decided the methodology definition and declaration to the specific purpose of software development through agile engineering methodologies. By this research author provided the architecture for internal and outside environment process.
24. **Alicia Raeburn [62],** studied suggested that agile methodologies framework depends on their characteristics and empirical evidence of studied and provided sufficient numbers of features and methods to adopt agility to accomplish the software development life cycle (SDLC) and given analytical framework of software designing.In this author point out the features of agile extreme programming (XP).
25. **Edmondson and McManus [63],** in their proposed framework main focus is give understanding and concern about decisions and statements made regarding the fixing and selection of agile method to accomplish the human centred software development (HCSD) and in here authors used a different case study in comparison on theoretical approaches to decide the agile engineering approach to complete the software development life cycle(SDLC).

2.4. PROJECT SCHEDULING THROUGH AGILE ENGINEERING :

26. **Yin [64],** the purpose of this study is given understanding about decision making in regard to choose of agile methodology in specific and tangible manner and for this author conduct case study over three different organization in three projects scheduling to each organizational scenario as shown below givne below table 2.1.

Table 2.1. Project Environment Description and Case Sites



Sl No	Company name*	No of employees	Turn over (₹)	Business focus	No of projects	Projects
1	ABC Limited	100,000 +	40 million	IT, BPO, Consulting	3	p1, p3, p5
2	DEF Technology Services	100,000 +	600 billion	IT, Consulting	1	p2
3	GHI Technologies	100		IT, Consulting	1	p4

2.5. SCRUM TECHNOLOGY :

27. **Mazzur, G [65],** have proposed data analysis framework for analysing data and values, in this they used four dimensions of system project scope, attributes and features, agility values, and designing process would be useful in get the degree of agility of software development through agile software engineering methods and in last they have identified influenced data values and extracted the data indexing and grading to conduct the data analysis and by their study they released analytical framework for the perspectives of project management, development, support and adaptability against agile methodologies. In their proposed framework they fix team size for XP agile technology and SCRUM technology is less than 10.
28. **Sriram Rajagopalan and Saji K Mathew [66],** given an analytical framework by which we ensure understanding how vendor and business stake holder decide the agile methodologies in software development and how it is fit in. By this proposed model authors first point out agile XP and their role in human centred software development and in second point they done analytical analysis about agile SCRUM technology and their significant use and scope for user adaptability and decide rule regulations excepted from software designing.
29. **According to B.A. Kitchenham and S. Charters [67],** In this authors done tertiary study that identified and derived catalogues features and individual SLRs in this significant software development research field and to achieve this authors done

tertiary study to get the dramatic impact and provide framework for industrial adoption and also work on technical mainframe layout for human centred soft development and review the other aspect of software development accordingly global software development context.

30. **According to Adel Alshamrani1 and Abdullah Bahattab [68],** the computer system and software engineering has become essential in today's modern era life, and it is widely accepted in many fields of today running environment and facilitate these work software development industries need software program and methods and parallelly software designing possible through some pre-defined development model: waterfall ,incremental ,spiral ,view model to accomplish the software development needs and objectives that are decided in starting of software development.
31. **Mihai Liviu DESPA [69],** given focus on the modern software development approaches and provided current state of knowledge in different software development methodologies and techniques and also provide the formalized software development approach dedicated to innovation field of software development industries and by this research author point out on specific characteristics of software development methodologies those managing software designing projects and enhance the capability project manager and in this research author, encountered conventional and modern technologies of software development and also by this research author formalizing a software designing approach and provided graphical representation of all innovation technologies of software development industries and discuss their merits and demerits also.
32. **Karthikeyan Chandran and Madhuchhandan Das Anudhe [70],** in this author given view on water fall methodology and discussed the features of first methodology of software development. Water fall is the globally accepted methodology, and provided acknowledgement to each phase of software development and dedicated to linear or sequential level of software development and its approach firstly developed and introduced by Winston W. Royce

2.6. CLEANROOM METHODLOGY :

33. **A. Spangler [71],** provide methodology that is called cleanroom methodology that is effectively used in software development. Cleanroom methodology mainly focuses on issue prevention because defect prevention cost and time is less expensive

as compare to removal the issues and defect. Cleanroom software development approach construct software without bug and issues and to accomplish this cleanroom software development approach used quality control by the using a development model based on mathematic and statistical testing model.

34. **M. Soeken, R. Wille and R. Drechsler [72],** have propose a behavior-driven software development(BSDM) methodology and this model based on user acceptance testing and in this model authors themself introduce and write requirement self-manner and test these requirement in the manner of acceptance testing and forward to the project owner's responses, feedback and constant constraints if required then add in development phase through the constant interaction and take place through the life cycle of different modules of software development phases.
35. **J. Bezivin [73],** has discussed on test-driven development approach of software development based on unit-testing approach or we can say work environment is unit testing development approach. In this methodology of software development software developer not write actual code or method in respect of this developer apply automated test cases mechanism and get the actual responses by end-users and business stake holder and tested the actual and expected result through feedback and this approach is helpful in medium and large size software project and newly functionality based software and this approach increased the legacy code environment in software development to achieved the quality standard in software development life cycle (SDLC).
36. **E. W. Duggana and C. S. Thachenkaryb [74],** have given view on joint application development [JAD] methodology core working is getting system requirements through end-user's interface and continuous feedback getting by management team and business stakeholder also and this activity highly recommended in requirement and designing phase of software developmentation and by this philosophy of software development joint application development [JAD] work started first on prototype in comparison to actual developmentation work. JAD methodology started meeting with project owners at every stage of project and their feedback before implement the actual prototype and joint application development [JAD] is used mainly suitable for small size of project.
37. **G. Madey, V. Freeh and R. Tynan [75],** have studied on open source software development (OSSD) model is used decentralized software development

methodology without any centralized control mechanism methodology, no direct project owner involvement and defies traditional software development methodology in those thousands of line of codes (KLOC) required and need needed more than thousands programmer and developers to write requirements specific codes and throughout software development life cycle most of open source code software developer never meet to each other face to face and so this reason there is no tuning come arise within development team. In open source software development (OSSD) methodology work on comprehensive software testing manner.

38. **G. Lory, D. Campbell, A. Robin, G. Simmons and P. Rytkonen [76],** have propose Microsoft solution framework for software development and work on defined set of objective principles, model, working environment principles. Microsoft solution framework proven practices for both version of software development lightweight and height weight application version level implementation policy, this approach basically applied in agile engineering software development approach by the use open communication and empowers team members empowers and increased accountability in through life cycle of software development. This approach of software development increases the team members efficiency and increased accountability and mixed portability.

39. **G. Pollice [77],** has given their view on rational unified process that provides a systematical approach to software development and generally this methodology given architectural road map of process designing for multiple types of software projects and provided guidance how to completed software project in specific time period and within me decided time frame. In this approach of software project development there is no involvement of project team in any kind of specific assigned work ana any specific activity and tasks too and this approach gives general framework or road through development team complete and commit their decided objectives and goal too.

2.7. HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD):

40. **Rajeev Sharma and Jitendra Nath Singh [78],** have given their view on design thinking (DT) and user’s requirement engineering in context of human centred software development (HCSD) and they point out that design thinking (DT)

approach decrease the technical dependency in software development life cycle (SDLC) on a particular and specific software development methodology and by this practice software project complexity falling down and gives optimize solution for end-users and the authors also provided significant view on the requirement engineering, it is a approach of software development by which we get the end-users feedback and capture their response and requirements expected form software and project and after the requirement specification approach we finalize the project actual requirement and demands and in last we have the actual and specific requirements for human centred software development (HCSD).

41. **Hein, Ganix and Ion [79],** have given view about human centred software development (HCSD) and said that HCSD as concept to involved to industrial development and point out that 20'th century economics is based on manufacturing industries and design thinking changed industries dimension scenario from manufacturing to production industries and final orientation is production based software development and involved user feedback and requirements in software development life cycle (SDLC).
42. **According to Omer,Uludag, Pascal, Putta,Maria and Casper [80],** have proposed framework based agile engineering for service dominant industry (SDI) that given multiple areas of service-oriented research and provide interface service design for service based specific applications and provide middle layer environment to accomplish the software development process. This suggestd framework save time and reduce data complexity in software process designing and by this approach the cost of software development can reduce and fulfill the customer requirement.
43. **Blomkvist and Holmid [81],** given view on industrial specific design phase to service design software development and attention overcome to process-driven and which effectively emphasize on human-centred software development(HCSD) and involved innovation aspects in prototyping development and help this approach by added creative ideas and thoughts and pointed out this development approach by more faster and convenient in compare to traditional process of software development and design thinking (DT) given fully service oriented approach to accomplish the user-oriented service designing.
44. **Liedtka, J.[82],** emphasis on design thinking (DT) and revolves throughout around the human centred software development (HCSD) at its central point and human centric involvement at its surface area and add more diverse set of heterogenous level

of software development and point out on diverse attributes of user level of point of view and increased the importance of co-design and co-creativity in software development accordingly to human centred development and also add bit of understanding involved with conduct a variety of ethnographic process research specific tools for complete process mapping, algorithm to achieving the objectives for a design thinking process development.

45. **Milena, Lalic,Maja, Jelena and Nenand [83],** have focus on agile and digital based products digitialand vitally accepted approach not for process development and also formulated the design process and often started the user's involvement and customer's feedback and ideas and enhance the customer's perception.

46. **Camacho [84],** has focus on iterative approach and non -linear methodology to promotes and point out the significant importance of quick prototype instead of board based structure which have drawbacks when simulation techniques apply on developed prototype and other hand design thinking (DT) achieves this objectives through iterative process prototype model in open development environment end for human centred software development (HCSD) and involved analytical design thinking to get the specific objectives according to user end.

47. **Dam and Siang [85],** have given problem solving features and attributes as hands on approach for problem solving environment as given below:
    - Empathize with operational level end-users
    - Declare and define needs and issues
    - Minimize challenges through ideate
    - Develop prototype
    - Apply testing and validated quality

48. **Clack, L., & Ellison and R [86],** have given research idea on co-designing and emphasis also on prototyping in design thinking (DT) technology because in this, authors critically examined that collaborative knowledge not only increased and generated data value insights for designing area of software development but also involved and allows for different opinions to developed robust product and also produce the user oriented product and application.

49. **Schumacher T, & Mayer S. [87],** have proposed research analysis on technological and functionality innovation, business development model, rapidly changes in customer behavior and requirements increasingly issues in and outside software

development environment and the success rate of software development dully depend to how mange these dynamics, and for this development manager and system analyst not work only to release the project through traditional problem oriented practices and techniques but also adopt alternative problem-solving and innovative design-centred problem solution to develop human-centric software .

50. **Anchit Shrivastava and Isha Jaggi [89],** In this research paper authors emphasize focus on agile engineering extreme programming (XP) approach and discuss their influences and execution innovation in business management and fulfill the end-user satisfaction and needs because it is more viable approach of software development in there quick changes and corresponding reacting responses required for the standard software development and also archived the business and stake holders pre-defined objectives and requirements too.

51. **Rajeev Sharma and J.N.Singh [166],** in this paper authors provides conceptual intelligence conceptual meta model for human-centric software designing process and process management and also increase user acceptace and statisfactions levels in through out the software development.
52. **Rajeev Sharma and J.N.Singh [167],** have processed reseach analysis for job statisfactions in respect of end-users requirements.In this reseach paper authors provides comparatives analysis in reference of agile-human centric and designing thinking (DT) for staic and dynamic requirements level and also discuss the reliability metrics creteria for human centred software development (HCSD).

2.8. CONCLUSION :

This session provides systematic literature review document (SRsD) studies of different approaches of software development and also finds the facts and figures by the different agile-human centric methodologies research papers based on extreme programming (XP) and sucrum technologies of agile egnieering and also analyse the uses of desing thinking (DT) in software development according to end-users environment by the systematic literature review documents (SRsD) studies. In the last of this chapter carry and figure out the human centred software development (HCSD) approaches features and their involvements in software development and process mangaemenets by the published different reseach papers.

CHAPTER - 3

RESEARCH METHODOLOGY

RESEARCH METHODOLOGY

This chapter impassively focus on research methodology and techniques to achieved and full-fill objectives and goals as decided in earlier stage of this research. This chapter provides empirical approaches and research mythologies to design and develop of integrated human centred software development (HCSD) model based on agile engineering and design thinking (DT) approaches. Agile engineering gives accelerated approach to software product delivery, software development process management, manage and control product priorities if arise at running stage of software development through user's end [89]. Design thinking (DT) is an approach that magnificent and increase the understanding of customer behaviors and needs from the process development and from end product delivery and make sure the technically and economically feasibility of software development [90]. This human centred software development (HCSD) developed by agile engineering extreme programming (XP), agile scrum approach with the integration of designing thinking (DT) approach to stratify the end-users requirements and needs and this conceptual metamodel gives user-driven and increase job satisfactions level in software development and process management to fulfill business stakeholders objectives and user-point of view.

3.1. AGILE ENGINEERING METHODOLOGIES:

The software designing activities becomes more complex due to the need and requirement of direct need of end-user participation in software development modeling and solutions to generate integrated software methodologies. Agile engineering generally divided requirements in multiple sprints/process activities for user statisfaction levels because in every phases of software development user feedback/responses involved to increase stakeholders acceptance index levels and environment according to end-operational stages. Agile engineering software development methodologies in glance totally acquire the software quality predication is advance and release different types of methodologies for user-oriented goals and requirements.

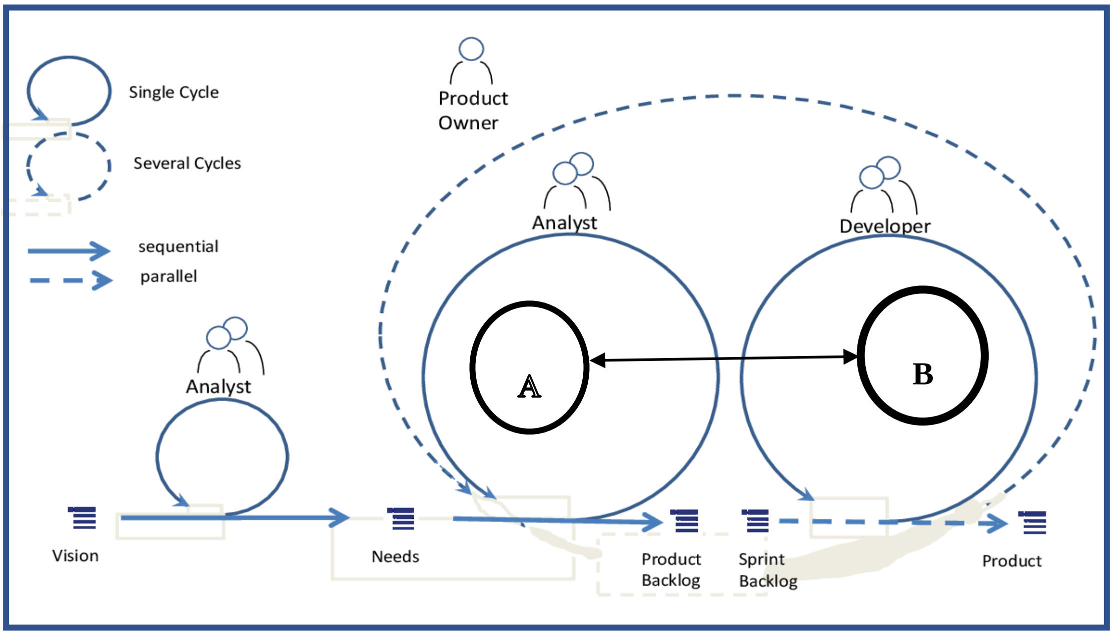


Figure 3.1. Agile Engineering working environment and development life cycle

Agile engineering has become more relevant, and increase product services [93] and provides organizational strategy and methodologies with an emphasis IT benefits and revenue growth of organizations and stakeholders [94-95] and release executive-based application for IT benefits [96]. Above figure 3.1 shows the agile engineering working environment and product life cycle and as in A and B showing point to point relationship manner and style in agile engineering methodologies between client and development end as shown in figure [91-92]. Agile engineering approaches aware the organizations in context to identify technologies applications based framework and recognized user's needs and expectations from the software development life cycle (SDLC) and secure the agility, and standard/quality oriented software development in simultaneously design style and manner. Agile engineering sometimes divided process in different pieces/sprints of process to gained the portability in different software development practices and techniques to ensure the quality of software and process designing. Agile engineering methodologies captures different user level job aspects and criteria to produce optimize process and business solutions according to human-centric solutions and needs. Main goal of agile engineering approaches are client satisfactions, portable business solutions, unique and job oriented business solutions, minimal uses of resources and lines of codes (LOC), reduce the code complexity, increase product efficiency and effectiveness by the uses of business process scripts/methods.

Agile engineering started by initial phase of process requirements to crystal clear approach/techniques to produce authentic and quality-oriented product.

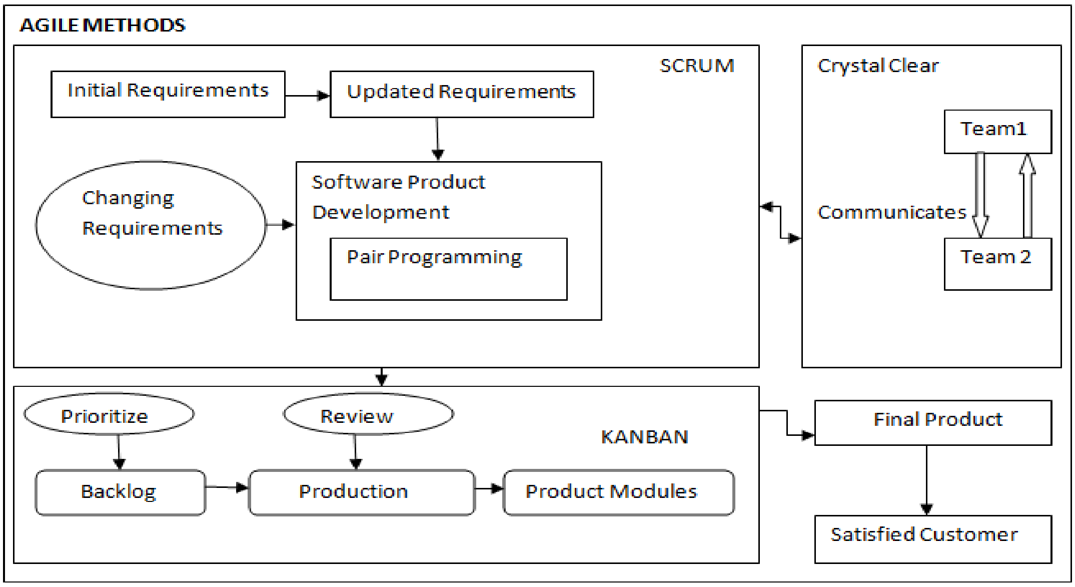


Figure: 3.2. Agile engineering working stages and its methodologies [91-92]

Agile engineering has unique philosophy, frameworks and different kind of approaches to solve development issues and objectives as shown in above figure 3.2 and in here given below point out different kind of agile engineering methodologies:

1. **ASD (Adaptive Software Development) Methodology**
2. **DSDM (Dynamic System Development Method) Methodology**
3. **AM (Agile Modeling) Methodology**
4. **XP (Extreme Programming) Methodology**
5. **Agile Crystal Methodology**
6. **FDD (Feature-Driven Development) Methodology**
7. **Scrum Methodology**
8. **ASP (Agile Software Process) Model Methodology**
9. **PP (Pragmatic Programming) Methodology**

3.1.1. Adaptive Software Development (ASD) Methodology [97]:

This is earlier stage of agile framework to develop outgrowth development framework and rapid application development (RAD) to satisfy the quickly and effectively changes in software development to achieve the objectives and goals of pre-decided software development stage. Adaptive software development (ASD) focus on overall project development, emphasis on the user-oriented self-dynamic teams, intercommunication in software development life cycle (SDLC) phases to developed successful, reliable and quality software product and below figure 3.3 shows the different stages of adaptive software development (ASD) life cycle and showing their processing level and relationship between all three phases: speculation, collaboration, and learing.

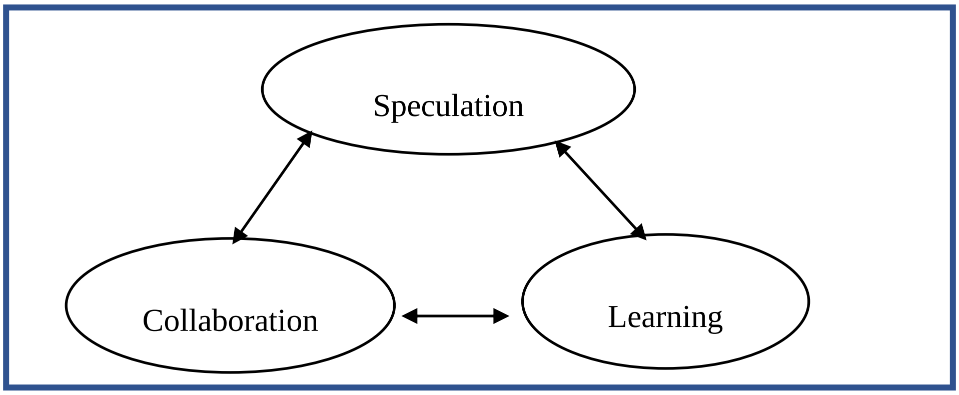


Figure 3.3 Adaptive software development (ASD) life cycle

Speculation:

In this phase of adaptive software development (ASD) life cycle project initialization started and then planning is conducted to achieved the development goal. Speculation phase done project plan and requirement initiation, specification, mission, objectives, and end-user's customer requirement statement and then release project development life cycle (SDLC) phases release for software development further progress. ASD methodology decided project mission, features, iterative innovation techniques involvement, time-bound scheduling, risk factors analysis and driven, and point-out future based changes tolerant in concern of software development [98].

Collaboration:

This phase built-up internal structure for communication and motivation for end-users and internal software development team members to increase feedback, motivation factors, collaborates team works and communication to increase creativity to individual level of software development to develop creativity and productivity of software product through healthy collaboration team environment where team may be point on but not animosity, helping environment within team members without any resentment feedback, adopt hard working environment to achieve objectives, always ready to adopt innovation idea and thoughts to accomplish assign work, and create communication environment within developing team and end-users to find out the best possible solution for problem.

Learning:

This phase gives availability to team members to think about the development methodology and if any changes and updations required in development environment then go on desired technology to increase the efficiency and effectiveness in software development life cycle. This level increases the development team understanding over the project development process and technology environment by increase team members focus, technical visibility within team members, and find out each and every fact about development methodology and techniques to develop quality-oriented software and product according to end-user's needs and requirements.

3.1.2. Dynamic System Development Method (DSDM) Methodology [99]:

Dynamic system development method (DSDM) methodology is iterative and cyclic incremental that core focuses on rapid delivery of process development and involve user throughout in software development life cycle (SDLC) and adopt dynamically approach in software development. DSDM agile engineering approach is used in both object-oriented designing and functional based software designing. Dynamic system development method (DSDM) methodology mainly in that environment where requirement is not fixed in advanced and in this entire team members work parallelly and simultaneously manner of software developmentas.This methodology involved user, improve decision integrity in every phase of software development, focus on recurrent level of all development phases and project delivery, iterative process development and accept dynamically changes in requirements if required in software development life cycle (SDLC). Dynamic system development method (DSDM) methodology start working on starting phase of software process designing and after fullfill the requirements according to verification and validation of end-users quality then after designing phase enter in next stage of software development.In dynamic system development method (DSDM) methodology time and resources work together to accomplish the decided functionality parameters and in this approach sometimes functionality become variable and play dynamic role in software development and process management.

Agile dynamic system development method (ADSD) methodology make synochronous between functionality and resources and encapsulate the variable of requirements as shown in given below figure 3.4 to make dynamic relationship in time, resouces, and functionality to increase effectiveness in dynamic stages of software development.

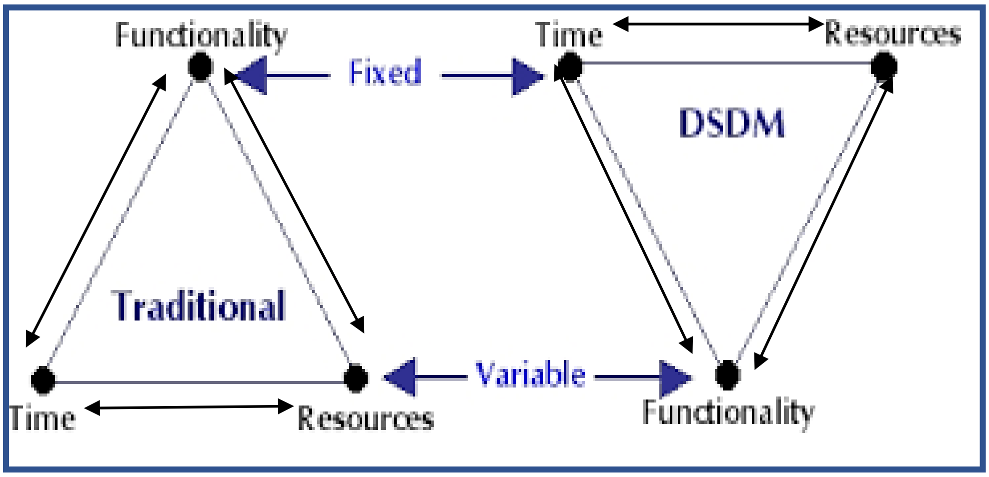


Figure 3.4. Traditional Vs DSDM Methodology Phases

3.1.3. Agile Modeling (AM) Methodology [100]:

Agile Modelling (AM) is methodology to develop software/product in acute requirement of software development through the agility and rapid development approach both merged here and this methodology not work individually but it works encapsulated manner of software development. Agile modelling (AM) is effective strategies for mapping and contribution of work through graphical manner documentation too. Agile modelling technique use two type of user interface in project development as discuss below.Agile modelling (AM) core working environment is on user sysntex interface level to end-user oriented level interface and for this this methodology main attributes is feedbacks and operational level responses in every stages of process development.

Command line interface: In this type of interface requirement of syntax is mandatory with system environment.

Graphical user interface: For this interface user used the graphical representation way to communicate with system.

By the above two core operational resoponses agile modeling (AM) provide syntax and graphical user communicate interface for project development and also work on rapid process development.Agile modeling (AM) techniques gives envision initial working architecture, model vision, user interface, communicate architecture to stakeholders and process designing team, develop response and feedback based architecture system, and in last ensure the end-users feedback on developer end as shown in below figure 3.5. Agile modelling (AM) focusing on mindset, people, disciplines of designing, modelling strategies, modelling sessions, scheduling the work, specification of work and discuss and requirements capture related techniques.

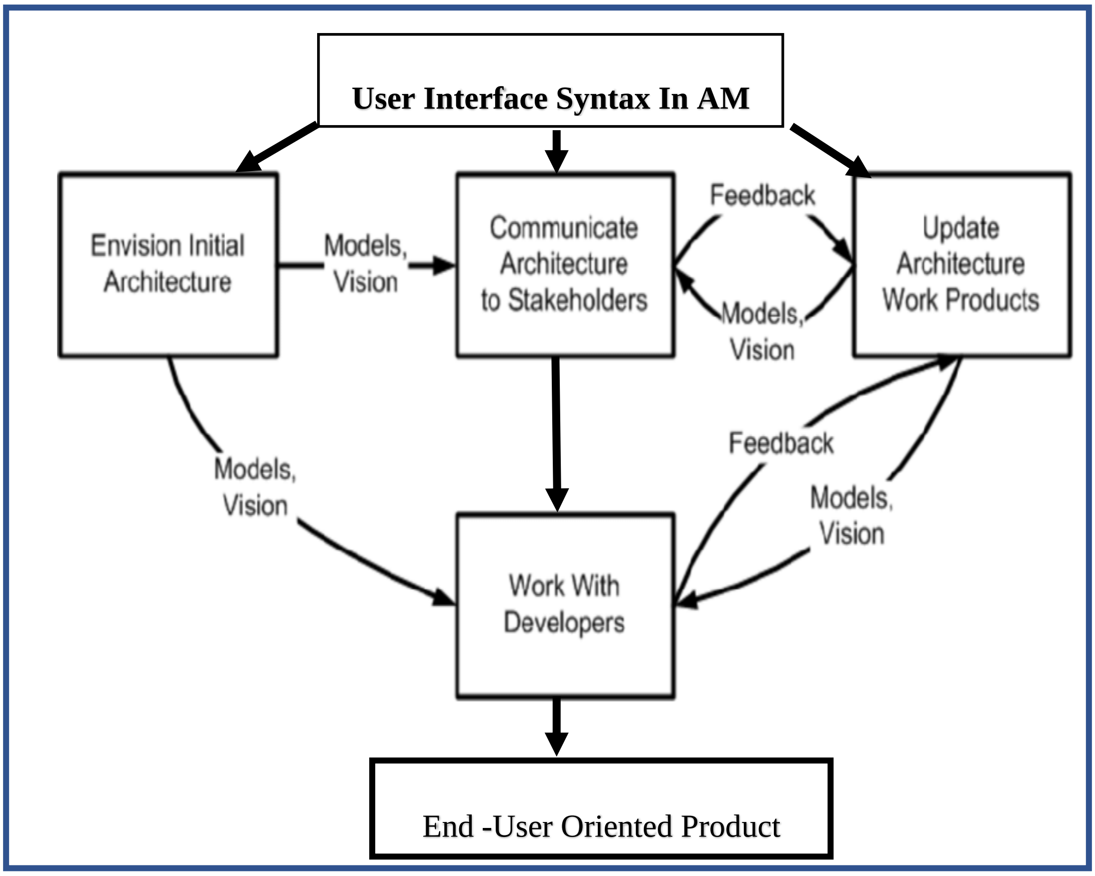

User Interface Syntax In AM
Envision Initial Architecture
Models, Vision
Communicate Architecture to Stakeholders
Feedback
Models, Vision
Update Architecture Work Products
Models, Vision
Feedback
Models, Vision
Work With Developers
End -User Oriented Product

Figure 3.5. Agile Modeling (AM) feedback communication architecture [91-92]

3.1.4. Extreme Programming (XP) Methodology :

Extreme programming (XP) was firstly introduced by Kent Beck in year 2000 and it is emerging agile methodology and it is offering a number of strategies, practices, attributes values and objective based principles which are suggested to be user-oriented application and magnified the software development process quiet easy manner [101]. Xtreme programming (XP) methodology suggested different thoughts and ideas of working as a package of serval aspect of innovative ideas and working principles by which software development process might be easier and convenient [102] and XP targeted generally co-located development teams in case of non-critical software development product and suggested to work for different sizes of industries world-wide [103-104]. Extreme programming (XP) provides a specific, simple and suggested native working principles and work around and guide the software development team throughout the main core phases of software development: requirement and planning, designing, coding and software quality assurance (SQA) and achieved goal and deliver the software product and application according to customer specific needs and within time bound limit and also ensure the software development because this approach has ability to acquire and accept new changes and requirement during the software development environment [105].

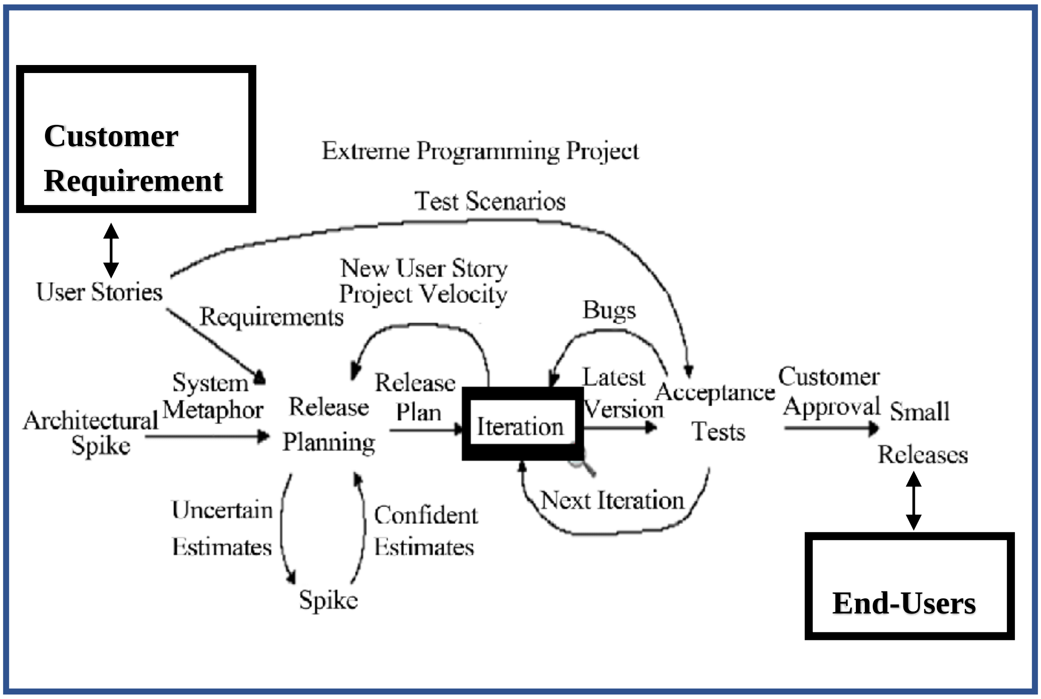


Figure 3.6. Extreme programming (XP) life cycle and customer feedback

This agile engineering techniques improves software designing process in four different phases of software designing : high level of communications between the developmet environment and end-users, light and simple design,real time modification according to feedback of last users end,and last one is core focus on actual requirement gathering through customer continuous feedback cycle [106] and above figure 3.6 shown the extreme progrmming (XP) life cycle and customer oriented feedback life cycle.This figure gives how the customer requirements passes through from different stages and phases in xtreme programming (XP) life cycle to develop customer statisfied software product.Extreme programming (XP) basically provides rules and practices, for the different phases of software development life cycle and these include: develop user stories,release small build of software design,reduce risk and increase availability within customer environment,pair programming, and optimzation uses of development resources and also full fill the standard of software development [105-106]. Extreme programming (XP) increasing effort,independent variable into core designing practices reduced the totoal designing efforts (dependent variable) and make the designing more specific needs of the end-users needs and accomplished what the stakeholders intended to do [107].

3.1.5. Agile Crystal Methodology :

Agile crystal methodologies were introduced by Alistar Cockburn in the year 2000 and they focused on increased efficiency and magnified habitability as components of software project safety [108] for this agile crystal methodology adopted certain rules and regulations, policy mechanism standards, as a tool for quality software product. Agile crystal methodology acquired the logic of crystal-clear approach for working on non-life critical software development approach and it focuses on people and not work on artifacts [109] and this approach focuses on designing team members, not development of artifacts [110-111]. Crystal clear practice of agile engineering is make an environment between people and communication such a way to developed transparent and optimally and healthy solutions for process designing [112]. Agile crystal development methodology the main essence behind this technique is release safe and end-user oriented working eco-system that can be easily changes requirement needs in development suite and below figure 3.7 show the different agile crystal methodology working style and their working cycle levels.

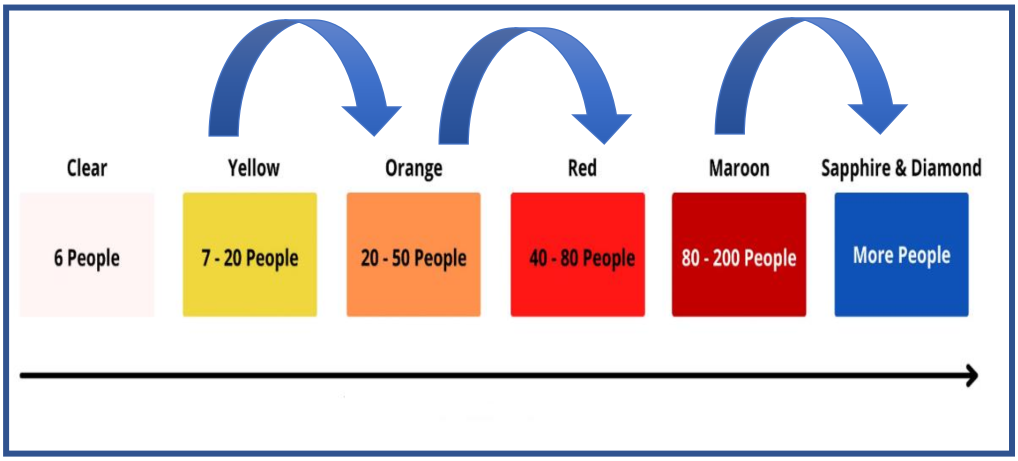


Figure 3.7. Agile crystal methodology working style [113]

3.1.6. Feature-Driven Development (FDD) Methodology:

This feature-driven development (FDD) methodology developed by Jeff De Luca and Peter Coad. This development methodology basically integrated some running software development strategies and methodologies in single environment [114] and feature-driven development (FDD) methodology quality determined from a client added valued functionality level viewpoint and focus on deliver tangible, and developed software repeatedly, dynamic manner and consider the timely. FDD methodology come across different plan and features as shown in below figure 3.8 and gives conceptual prospective of feature driven methodology (FDD) and also give overview about the different working model distribution level/plan for developing software product and application.

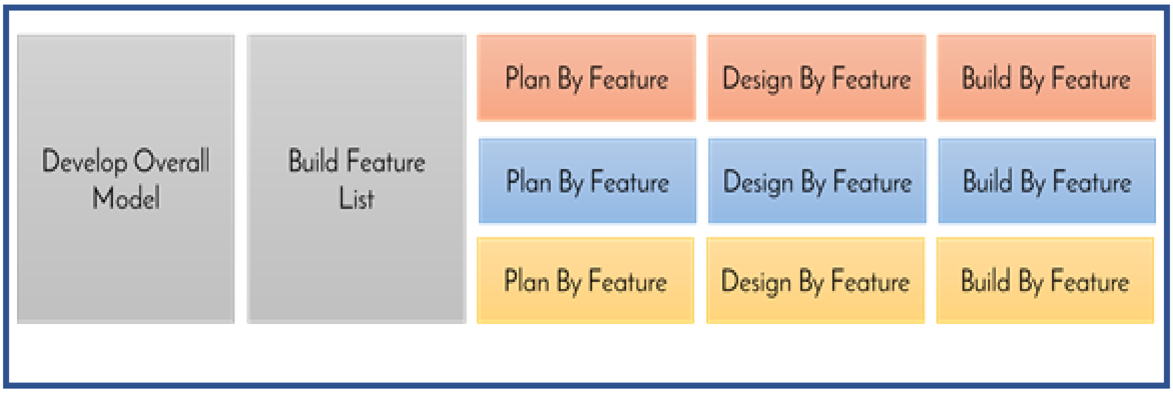


Figure: 3.8. Feature driven development methodology (FDD) methodology phases

3.1.7. Scrum Methodology [115] :

Scrum methodology developed/initiated by Ken Swaber in last year 1995 and this technology get identification in the IT industry before the release of agile manifesto and afterward this methodology come under the umbrella framework of agile engineering methodologies framework because it has same and unique features underlying come across as mentioned and describe in agile manifesto technology framework. Scrum technology generally used development process model that can be based on “zero sprint pattern solution”, and “one sprint ahead style” and the “parallel track pattern” and combined ideas and process in single platform [116]. Scrum technology main underlying concept is used simplifying process designing and project management through realistic processes, and application portable approach of process development and fundamentally focus on higher intercommunication within team environment [117]. The core mechanism of SCRUM is to build a process backlog and for this backlog is an area where designing team look after and see all requirements pending for a project scheduling ,sized based on complexity and decided team measurement and release simple requirement and sentence for each requirement and also point out discussions and declared what is needed to be implemented by the software development team to accomplishment the requirements

[118] as shown in below figure 3.9 Scrum methodology become acceptable and increasingly prevalent in software process development because it develop/creates a method/logics to successfully integrated with loosely defined and dynamic requirement changing environments [119].

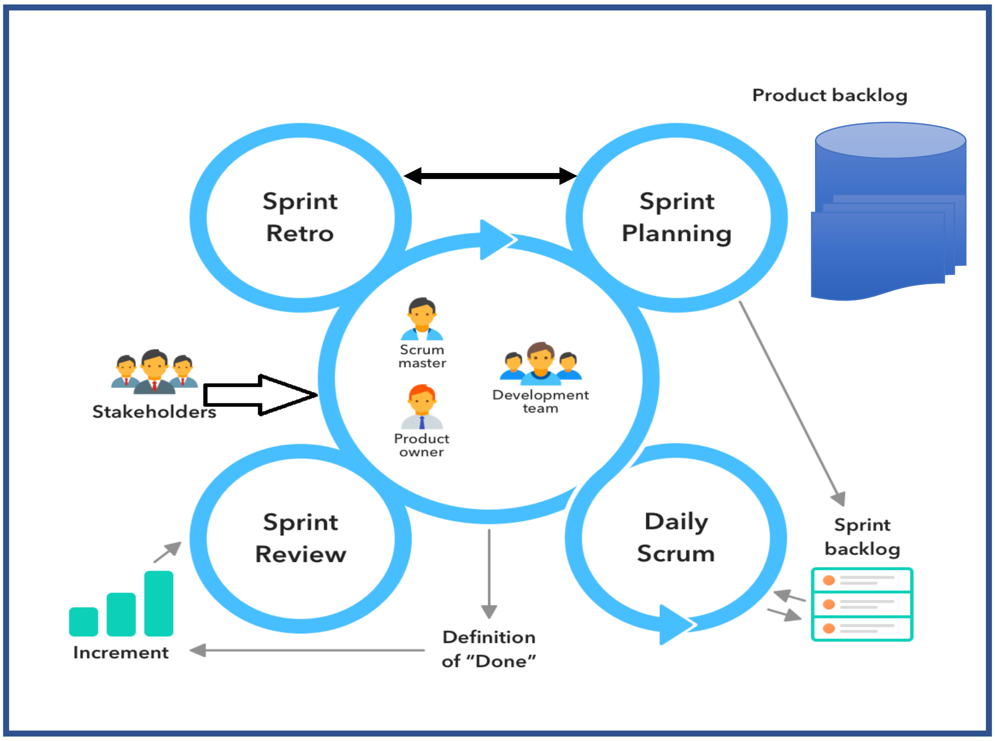


Figure 3.9. Scrum technology architecture

Software development through "scrum technology core aspect is divides the project into different phases of sprint" and each phases in Scrum methodology is fully developed with unique functionality, tested and be ready for production till the final implementation of project and satisfy end user's needs/objectives [120].Tradition software development fails to address usability requirements and needs of end-users because project stakeholders and owners generally not gives focus and attentions on project usability and their main focus on business issues and objectives [121-122]. Traditional agile methodologies not much concerned about the user expectation, vision, and not have specific architecture and working environment to capture, examined and ensuring the user experiences and needs in respect to project development and briefly scrum technology is considered as, iterative, prototype based incremental software development approach for project [123].

3.1.8. Agile Software Process Model (ASPM) Methodology:

This Approach believe that project needs to be handled differently approach to resolved project requirements through the best suite solution and in this methodology, tasks are divided in small time box frame for specific requirements and features to be release. Agile software process model (ASPM) methodology mainly work on iterative approach and manner to develop software and hold the customer requirements in each and every phase of software development and to achieved this "agile software process model (ASPM) methodology has four significant features: individuals and interactions, continuously software development approach, customer collaboration and communication, and be ready to changing in software development environment if any quick changes arise in software development phase by end-users" [124].

3.1.9. Pragmatic Programming Agile Engineering Approach:

In general, agile pragmatic programming is called disciplined agile and it is combination of waterfall and scrum technology and it also provides opportunity to control, manage various attributes/actions, such as procurement, business architecture, financial issues, IT management prospective, and provide convenient and specific solution in productive manner as shown in givne below figure 3.10 shwon the framework of pragmatic programming.

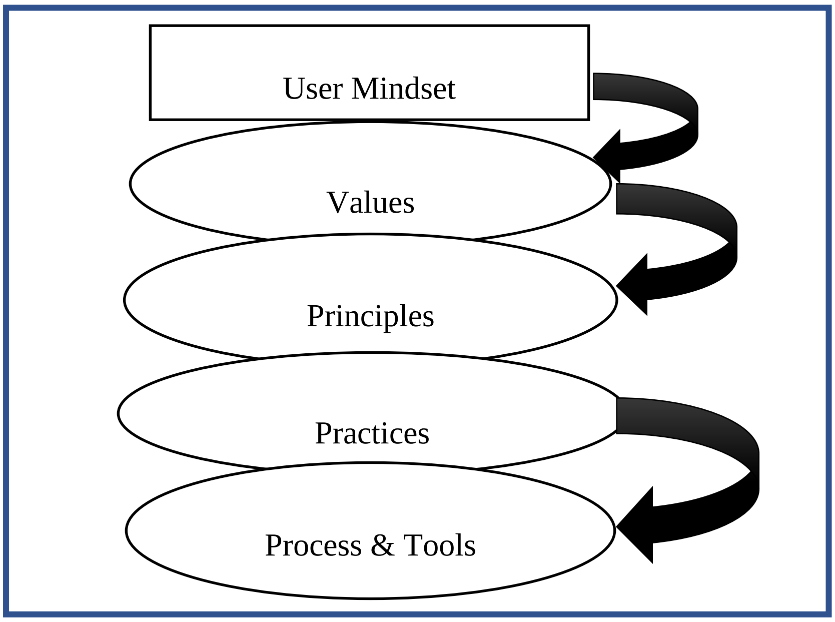


Figure 3.10. Pragmatic agile engineering framework

“Pragmatic programming agile engineering categorized in four different edges of software development: disciplined agile delivery O.x, disciplined agile delivery 1.x, disciplined agile delivery 2.x, disciplined agile delivery 3.x” [125]. Pragmatic programming reduce the data complexity and increase dynamic requirments acceptance from user end in middele phase of designing. Agile pragmatic programming methodology provides effective and suitable solution for operations optimization in entire work flow in business environment and working environment of agile pragmatic programming approach and their working attributes used to develop software according to the rapid changes, if arise in the running stage of software development through the user mindset.

3.2. DESIGN THINKING (DT) WORKING ENVRIONMENT AND FEATURES [126-128]:

Design thinking (DT) is an approach to deliver software product and services grounded in a human-centred development process and developed product inspired, motivated, committed always towards the end-user’s mindset and satisfactions too. Achieving “embracing software development quality acquired and understanding of users’ requirements and desires as well their operational environmental context”, and match these understanding with the process development practice and software development life cycle (SDLC) and work on “voice of the human centred approach”. Design thinking (DT) is work on convergence functionality that emerges from other emergent attributes of software development qualities, such as process reliability, controlling and monitoring, security, privacy, usability and maintain development quality index and performance level also. Design thinking (DT) core focus area is a user centred design approach and critical objective is examined user needs, objectives and impediments as point out to what is really being designed. Design thinking (DT) adopt ‘lean’ software designing methodology for the purpose of software development and business centric concepts of minimal marketable feature (MMF) and minimal viable product (MVP) for the resource’s definition and minimizations of time and investment. Design thinking (DT) encapsulate innovation to ideation and then by the creativity make different solutions for different service coustomers and deploy immersive, empathentic co-design process environment suitable and validated product development. Design thinking (DT) is an innovative approach for human-centred methodology that has achieved visibility and importance for its vital efficacy and efficiency growth in generating and quality testing innovative thoughts of software development process and techniques [129] and its different process solution shown in below figure 3.11. The results obtained by design thinking (DT) approach can be considered positive solution up to 95% completeness of process development and 100% of stakeholders' business requirements and objectives [130].

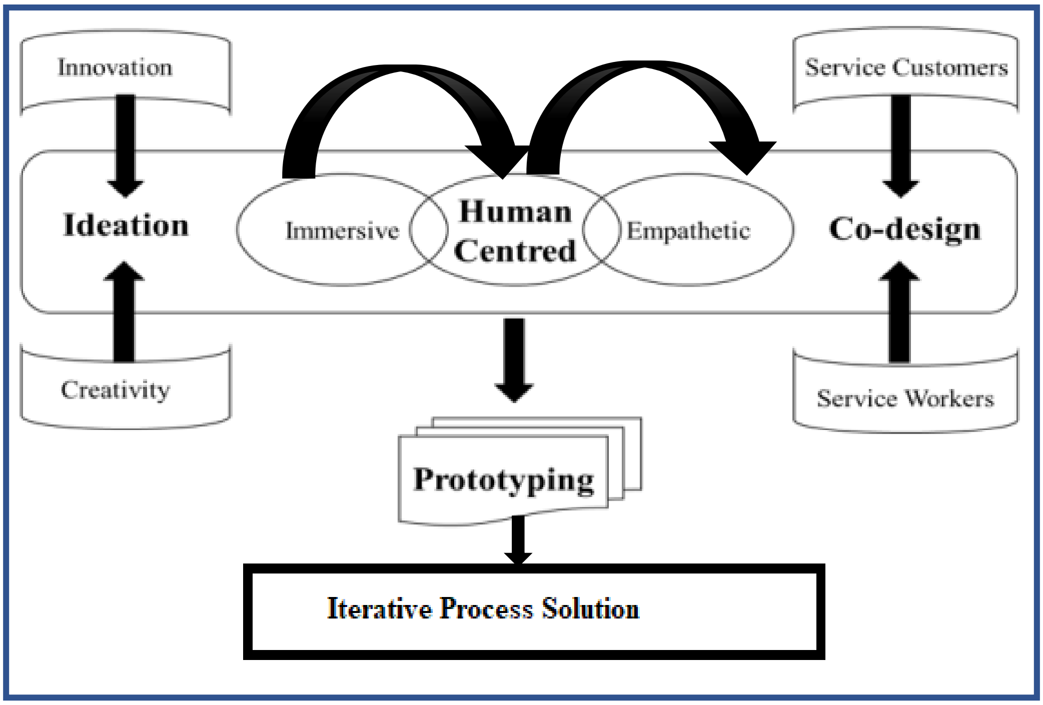


Figure 3.11. Design thinking (DT) working environment and iterative process solution for human centred software development (HCSD)

Design thinking (DT) working environment depends on five core attributes as discuss below:

- Empathize
- Define
- Ideate
- Prototype
- Test

Empathize: Empathize is the first step towards the software development through design thinking (DT) because it is core skill which allows designing team to understand, share the same thoughts as the feeling by end-user's in their mind. Empathize the first of design thinking (DT), where software development team real insight into users and their needs and in the focus on human-centric approach through consult experts to find out more conducting and concerning solution and observation of users feedback and gain a deeper, personal understanding of user's physical environment issues and takes their experiences and motivations.

Define: This phase of design thinking (DT) emphasize and focus on user's point-of-view input/statement and also synthesis/analysis, problem issues and statement according the end-result and give platform for human centric development and in brief user's concern and acceptance form development end .In this phase of design thinking (DT) we started to accumulate the resource/requirements gathered in empathize stage and work on observations and begin realistic definition of core problem for further development on that and provide the "real/analyzed problem statement of users to development team".

Ideate: Point of view analysis, design problem template, brainstorming, provides solid background through quality ideas/innovation and "think outside the traditional approach of designing" and give alternative and dynamic solution of problem. In phase design thinking (DT) designer team work on real challenges of assumptions and create potential and portable designing solution towards the user's mind-set or point-of -view.

Prototype: Design thinking (DT) this phase provides sketching, non-functional and functional prototyping solutions and development model ,storyboard, and start to create solution different solutions and started experiment on prototyping solutions and scale down version of product to investigate them on end-user's specific requirement and needs and also "identify the best and specific, quality oriented possible solution and prototyping model of designing" among the multiple solutions and prototyping of designing.

Test: This phase provides user's feedback grid/matrix ,minimum viable solution/product rollout from the development phase, and provides best solutions come out from the specific parameters have been put out on prescribed prototyping solution and match the expected and actual output to takes a mind-set of users and those requirement set in initial phase and level of previous stages of designing and development. Design thinking (DT) single line data process statement shown in given below figure 3.12 that mainly emphasize on end-users satisfaction and result and release real time operational product sketch in very beginning stage of software development life cycle (SDLC).

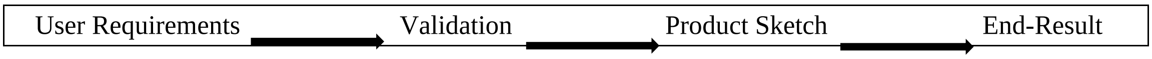


Figure 3.12. Single line data process statement of design thinking (DT)

so we can say “this strategy of design thinking (DT) approach give standard and solutions according to enn-operational users and provide essential platform for human centred software development (HCSD). Design thinking (DT) provide graphical prototype solution on the basis of expected outcomes and result and in below figure 3.13 shown the life cycle phases of design thinking (DT) approach and their relationship in concern of user acceptance index factors.

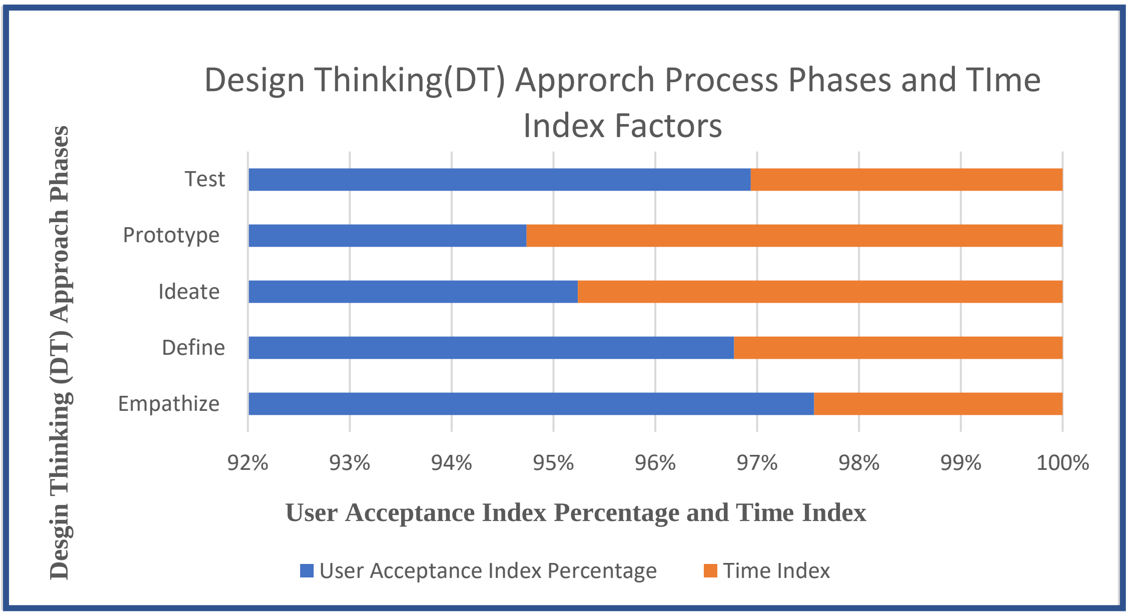


Figure 3.13. Design thinking (DT) core components and their role in user acceptance and time index

3.3. INTEGRATION OF AGILE-HUMAN CENTRIC AND DESIGN THINKING (DT) METHODOLOGIES FOR HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) [131-133]:

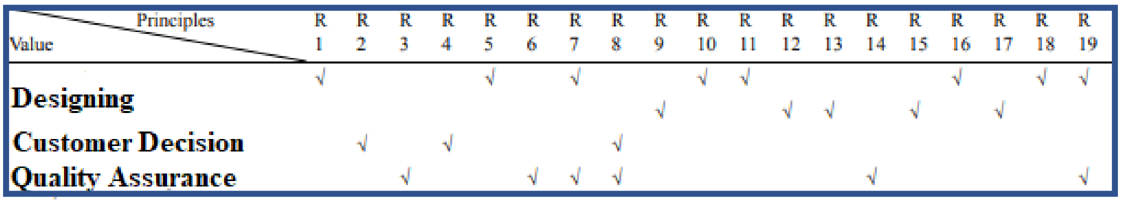
The major concern and issue to develop software with higher level of usability and it is seeming to quite so difficult to apply human centred design process to software development process and designing. To solve this, we propose approach basis on some factors:

1) Clarification of “user requirement/demand” and evaluation approach/process to achieved these requirements.
2) Fill the gaps between software development life cycle (SDLC), software engineering methodologies and user usability experts on the basis of human centred software development (HCSD).

3) Give solution of twin high problem (gap between end-user's requirements, problem solution algorithm and working architecture).
4) Increase the process durability according to end-user's satisfaction and also improved the efficiency.
5) Empowered to make decisions and the main concern is on frequent delivery of product enhancement.
6) Suitable business solution in order to converge on accurate business solution and also concern on reversable changed in development process.

So, these issues of software development can't be capture by agile engineering and design thinking (DT) alone and to developed human centred software development might be easier from the integration of agile engineering and design thinking (DT). In this development approach we introduced the scaled agile framework for business environment (SAFBE), which facilitates and work on planning and coordination across scrum working environment throughout the software development life cycle (SDLC) with the help of design thinking (DT). In proposed research methodology one technique of agile engineering is "scrum technology" that is suitable to provide suitable structured architectural framework for the rigid and human centred software development (HCSD) and also for complex product development. The main aspects and goals of the proposed research solution mainly core area is provide optimize communication between team members of development team and through design thinking (DT) to reduce the dependencies between the different area of software development as designing, customer decision, and quality assurances as shown in below table 3.1 different designing factors, customer decision and quality assurance scaled factors those are denotes as tick sign.

Table 3.1. Designing factors, customer decision and quality assurance scaled factors through integration of agile engineering and design thinking (DT) approaches



Principles Value	R 1	R 2	R 3	R 4	R 5	R 6	R 7	R 8	R 9	R 10	R 11	R 12	R 13	R 14	R 15	R 16	R 17	R 18	R 19
Designing	√				√		√			√	√					√		√	√
									√			√	√		√		√		
Customer Decision		√		√				√											
Quality Assurance			√			√	√	√						√					√

Integration of agile-huam centric and design thinking approach provide sophisticated, realiable, extensive, and provide closer solution to customers statisfaction and ensure product working behaviour and operational criteria.

Below table 3.2 point out research methodology and their core area, key issues and level of software development towards the human centred satisfaction and quality oriented and some core fundamental areas are agility, prototype - inherit from desing thinking (DT), scripts, and extreme programming.

Table 3.2. Research methodology approaches core area

S. No	Proposed Approach	Core Area	Key Issues	Level of Software Development
1.	Agile engineering	Agility	Problem Finding	End level user identification
2.	Design thinking (DT)	Prototype	Create prototype	Develop prototype at design level
3.	Agile Scrum	Scripts	Fragmentation of project	Divided task in different scrum for creativity
4.	Agile XP	Extreme programming	Provide new solution for user end	Gives solution as user prospective and satisfaction

So, by the above research methodology it calls for a high potential degree of empathy and understanding of end user and iterative process development new ideas, and makes possible solution for human centred software development (HCSD) and provides assumptions and redefining input problems, and provides alternative solutions in middle of software development. This proposed research methodologies gives/build solutions that solve user centred problem and seeking frequent input from end-users in order to develop iterate to the right outputs and outcomes, creates cross functional teams, balance design and development and in last for problem finding and problem solving for better outcomes towards the human centred software development (HCSD).

3.4. RESEARCH METHODOLOGY FOR DESIGN AND DEVELOP HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL:

In this session we discuss, core research methodology for “desing and develop human centred software development (HCSD) conceptual metamodel” depends on integrations of agile-human centric and design thinking (DT) approaches.The integrated approach mainly focus on to complete the end-users operational behavioural point of view and statisfactions level from the developed product and applications. This human centred software development (HCSD) conceptual metamodel integrated stepes of research methodology are discuss below:

1. First user requirements to file storage to chunk process and filter data chunk generation attributes through agile engineering XP & Scrum.
2. In here, conceptual human centred software development model used design thinking (DT) approach to filter data chunk generation for user storage & indexing.
3. Conceptual human centred software development used file container for user requirements to develop file 1 and file 2.
4. Conceptual human centred software development then integrate the agile & design thinking (DT) approach for develop dynamic software.
5. This develop conceptual human centred software development model finally integrate and validate the user's requirements.
6. In last stage of algorithm is produce blueprint of user requirement and designing platform and satisfy end-user requirements in middle of software development life cycle (SDLC).

Agile engineering (XP, Scrum) and design thinking (DT) approaches integration interface is data chunk generation point as shown in below figure 3.14 also point out different file container levels (File 1…File 2…).

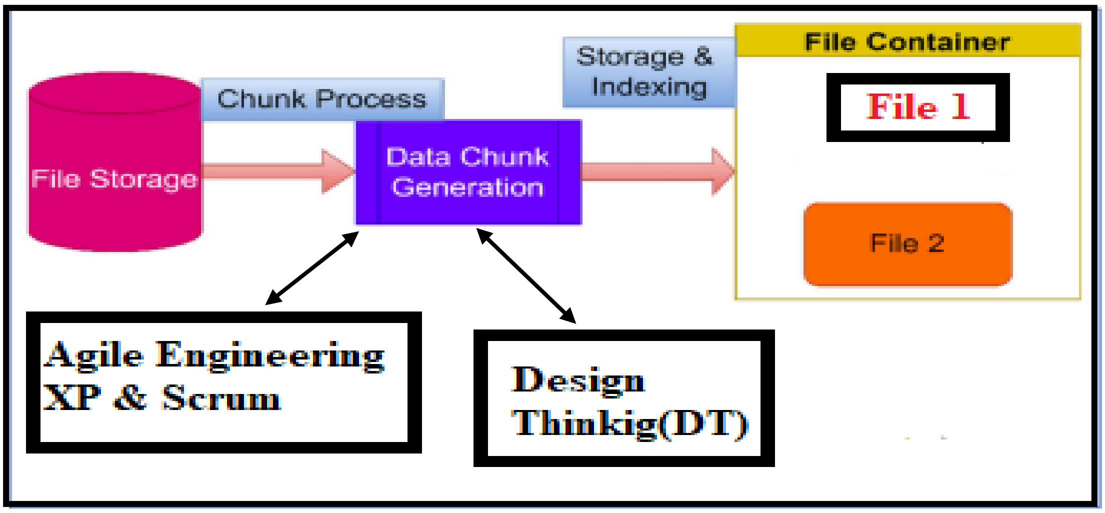


Figure 3.14. Software process designing by the integration of Agile Engineering Scrum & XP and Design Thinking (DT)

3.5. CONCLUSION :

This chapter provides the reserch methodology and way to design and develop human centred software development (HCSC) conceptual metamodel for process management and designing.In this session of reseach,discuss the integration of agile-human centric and design thinking (DT) stages of integrations for the human centred software development (HCSD) and for the agile-human centric extreme programming (XP), scrum and design thinking (DT) approaches are used to develop conceptual metamodel for process management and desging environment.

CHAPTER - 4

HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL

HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL

Design and develop conceptual integrated process management metamodel framework and its working functionalities are depending on agile human-centric engineering and design thinking (DT) approach and for this we integrated the agile engineering extended programming (XP), and scrum technology techniques with the design thinking (DT) approaches to develop the software for human centred point of view. In this chapter first we discuss the comparative analysis on agile engineering and design thinking and then we discuss the features of conceptual metamodel, steps to designed metamodel and framework and architecture to human centred software development (HCSD) model and point out how this model give more effective and productive outcomes for end-user point of view and also achieved the business stakeholder interests and objectives.

4.1. DEVELOP HCSD CONCEPTUAL METAMODEL FEATURES :

This model core working is “integration of agile engineering approaches and design thinking (DT) techniques to develop human centred software” and main feature is increasing the satisfaction of end-user’s requirements and demands and within time limit and optimize solution of resources and scheduling of project. Develop human centred software development (HCSD) conceptual metamodel major features are listed below [132-140]:

- Satisfaction level of end-user and project stakeholder at very high level
- Involved customer responses and feedback within software development life Cycle and no rigid approach of development use in this conceptual development model
- Dynamic working environment and involved the real innovation in project development and Time constraints and resource allocation in limit
- Collaboration and communication within team very high level on every stages of project development and project delivery and execution in efficient manner
- Issues and bug isolation can be handled very effectively after post implementation phase of software and product
- Scheduling of jobs and time frame built up is very easy and convenient

- Agile engineering approaches “Scrum and Extreme Programming” play significant role to designing this conceptual model.
- Conceptual model used “Design Thinking (DT) user’s involvement technique” almost every stages of software project developmentation [141-145].

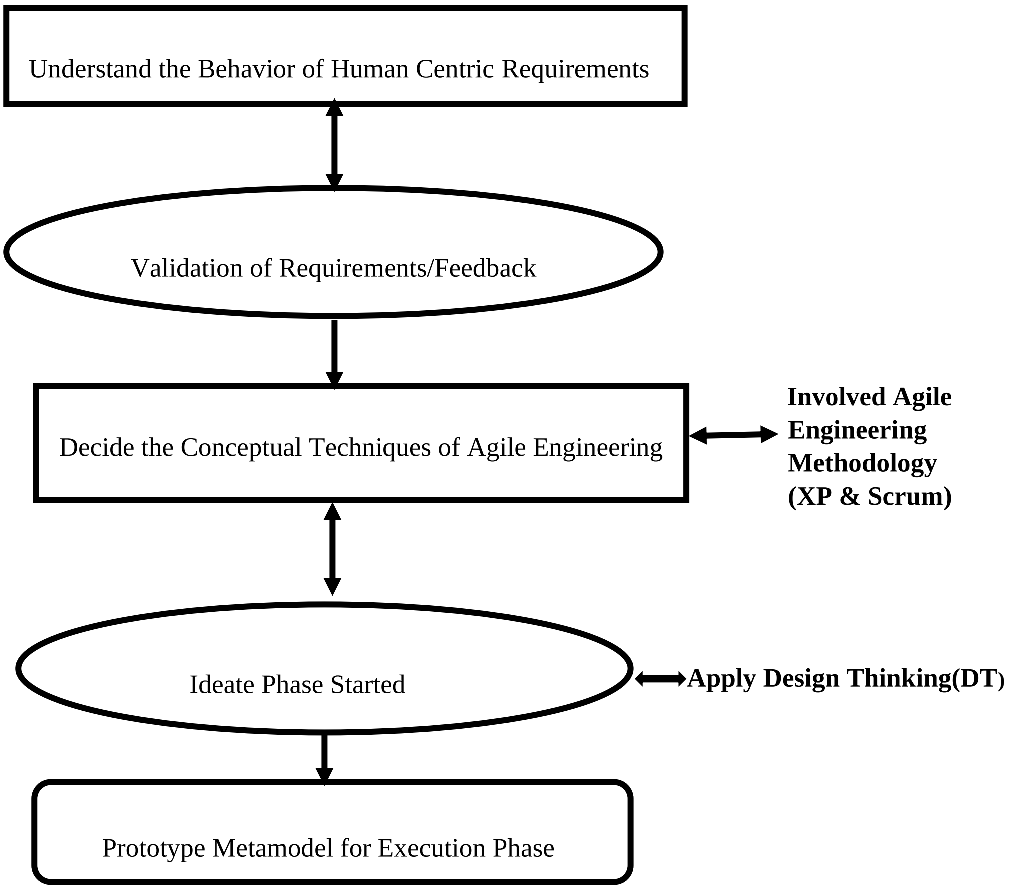

Understand the Behavior of Human Centric Requirements
Validation of Requirements/Feedback
Decide the Conceptual Techniques of Agile Engineering
Involved Agile Engineering Methodology (XP & Scrum)
Ideate Phase Started
Apply Design Thinking(DT)
Prototype Metamodel for Execution Phase

Figure 4.1. Develop Human Centred Software Development (HCSD) Conceptual Metamodel Different Phases

Different features of develop human centred software development (HCSD) conceptual metamodel and their relationship to others interdisciplinary phases of software development shown in above figure 4.1. This software development model firstly understands the behavior of the requirements and end-user’s expectation from the software development. Then requirements going for verification and validation phase. Then in third phase the where we decided the which techniques of agile engineering are suitable to produce sustainable software according to human centred approach. After the selection of conceptual approach of agile engineering for develop human centred software development (HCSD) then after apply the design thinking (DT) approaches for designing environment and finalize the ideate phase and completion of ideate phase now the system prototype enters in execution phase [156-160] and ready for end-user operational level.

4.2. ANNOTATION TRANSFORMATION IN HCSD CONCEPTUAL METAMODEL:

Develop conceptual process management metamodel is a human-centred endeavor and to achieved this we selected the suitable agile-human centric engineering and design thinking (DT) approaches to designed conceptual human centred software development (HCSD) for process management and development. The software develops directly accessible by end-user for operational point of view and feedback and in here human-centric approach determine the efficacy of software/product completed by software development team [162]. So this is issue become the idea and platform to designed and develop this human-centred software development (HCSD) and for this reason we analyze the user experience and for this product designers leverage their expertise and focus on understanding human-centred requirements effectiveness, interfaces and specifications because application model are only as effective if their users find them to be acceptable and useful otherwise human-centric issue/error is actually bad designing aspects and arise question mark on product creditability. In this model, development started by inspiration and ideation of user's requirements, get feedback by users through communication and feedback, involved users in real-time of project and also during the process designing and analyze the architecture behavior through the user-oriented process development and project management. In below 4.2 shown the different software development stages and orders of process management and development levels.

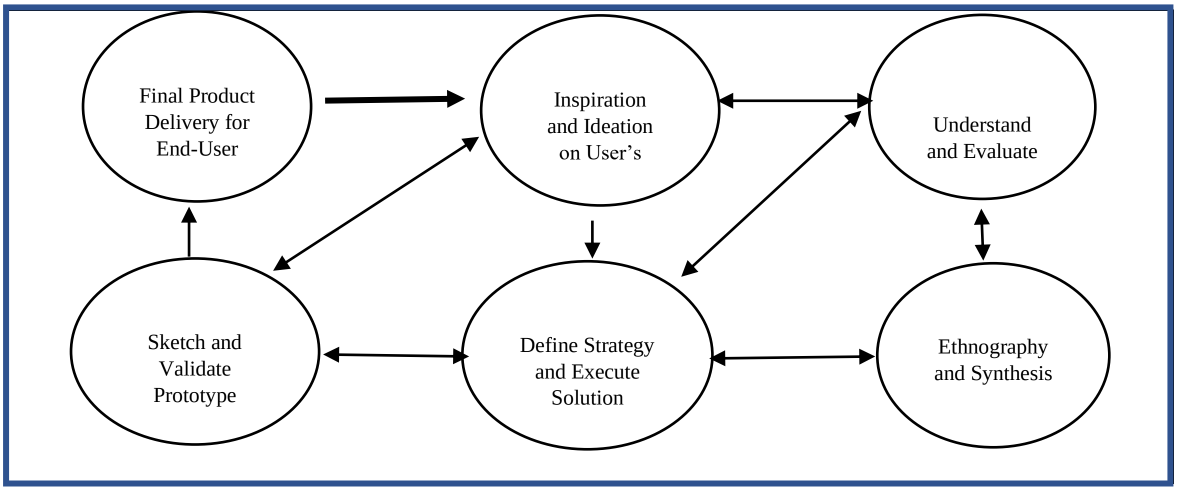


Figure 4.2. Conceptual Metamodel stages and their orders of process designing

The conceptual metamodel integrates the ideas/requirements of agile human-centric approaches into design thinking (DT) to know the existing software development life cycle (SDLC) to known process patterns to get the combines advantages of HCD and Scrum and XP approaches. In this process management metamodel end-user inspiration and ideation for user's requirements and understand and evaluate prototype-based solution and plan for execute strategy and ethnography and synthesis, validation and ethnography for end-human centric environment.

4.3. DEVELOP HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL PROCESS MANAGEMENT METRICS ANALYSIS:

In this section, we present the flexibility of different process and designing activities of human centred software development (HCSD) process management metamodel. The Conceptual integrated process management metamodel has been come in existence by performing fallowing steps as discuss in below [163]:

Step 1. Parsers of methodologies:

For this step 1, we adopt model driven reverse engineering (MDRE) approach delivered language and technology based metamodel from the source technologies and traditional agile based human centric approaches.

Step 2. Architectural metamodel transformation of technologies:

In this step 2, we decided structural elements values and attributes of metamodel that final end-user's requirements and decisions point of view and aspects.

Step 3. Metric transformation and technologies annotations process:

Developed conceptual human centred software development (HCSD) model gained and give quality of success in process management by order of three stages:

1). Metrics transformation developmentation steps to achieved and measured the output of architectural metamodel. In this stage of metric transformation, we decided the methodology for conceptual human-centred software development (HCSD).

2). Annotation of the methodology's architecture metamodel here we decided technology for metamodel and give classify the suitable and strong technology decencies for conceptual framework. In this stage of annotation of mythology takes the advantages of agile-human centric XP and Scrum technologies and in sequentially manner of annotation of technology enhancement and process management we integrated here design thinking (DT) approach to satisfactions and quality of end-user operational environment.

3). Human-centred technology-independent level of specification (HCTILS) metamodel in here we decided intermediated technology architecture of agile-human centric and design thinking (DT) approaches that allows significant and concrete graphical level user-oriented framework. Conceptual human centred software development (HCSD) model used some "metrics transformation" to increase productivity and effectiveness in software development and some parameters of metrics transformation are discuss below [160-163]:

- **Metric system:** Discuss the user requirements and specification standards
- **Measurement:** Conceptually integrate through and release for measurement
- **Creating metric model:** In this transformation level we create metric model for human-centric requirement specification and analysis for this metric model takes human-centric observation and innovation factors of process designing.
- **Develop user querying architecture:** In this stage of metric transformation we suggest and passed requirements/queries of end-user to developed process management metamodel to assurance of requirements and outcomes. In here we decided process delivery and designing system on the basis of end-user's scenario and domain. In user querying architecture model, in this we explicitly reduce process complexity by the maximum involvement of user interaction and motivation factors.
- **Human-centric structured metric metamodel (HCSMM) :** In this level of metric transformation we provide human-centric structure metamodel (HCSMM) for test the actual and expected outcomes of developed model.In this research we designed and develop process management metamodel based on agile-human centric and design thinking (DT) approaches as mentioned above in previous section that the core and typical focus area of develop human centred software development model (HCSD) and its phases and stages are managed dynamic requirements during the different phases of software development and business

stake holder always demand product delivery frequently manner and user oriented. So traditional development approaches not offer secure process delivery thus this conceptual development model main secure focus area of designing are discussed below:

- Agile software development (ASD) approaches always focus and working on software success and product delivery and not emphasize and pay attention to the human-centric secure delivery in process designing.
- User's involvement in secure software delivery is always essential and mandatory and we involved these parameters in starting phases of software designing.
- By the used of design thinking (DT) while adopting/deciding software development methodology for human-centric software development by the involvement of end- user's feedback and responses and implementation of software is always human-centric.
- In this derived conceptual metamodel we inherit the properties of agile-human centric approaches e.g. agile engineering extreme programming (XP) and agile engineering scrum approach and parallelly involved the design- thinking (DT) different process stages like: empathize, define, ideate, prototype, and test.
- This conceptual metamodel work on conceptual structure metrics metamodel (CSMM) and gives graphical representation of source process and adopt express architectural framework of software designing process and track architectural changes, compliance with specify understandability for reference metrics architecture so that end-users can test the visualization develop software and process designing.
- Develop human centred software development (HCSD) provide well defined metric reliability approaches, query based analysis phases.
- Operential level flexibity section to accept customer feedback to enhance the process effectiveness
- Actively involved data integrity, quantifiable approach and well defined artifcats for software development and designing and also increase reliability and scalability for process and methods.
- Develop conceptual metamodel uses different kinds of metrics annotation for multiple type project handling and process verification and validation.

Various metric process management statement depicted as shown in below figure 4.3 and gives the different metrics transformation and annotation stages of developed human-centred software development metamodel and depicted the metrics transformation stages of developed human-centred software development (HCSD) and their different working flow levels

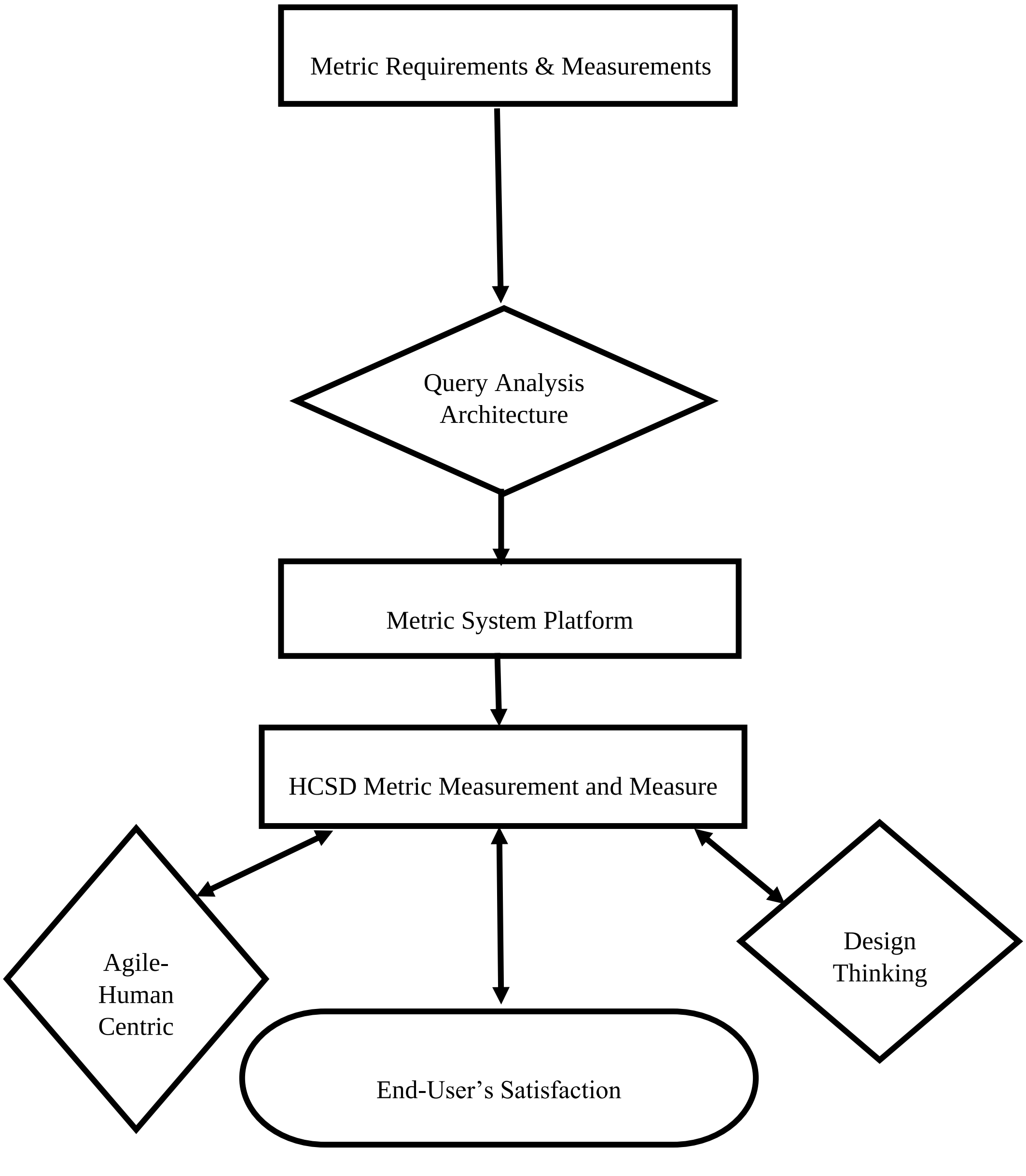

Metric Requirements & Measurements
Query Analysis Architecture
Metric System Platform
HCSD Metric Measurement and Measure
Agile-Human Centric
End-User's Satisfaction
Design Thinking

Figure 4.3. Metrics transformation and measurement stages in develop human centred software development (HCSD) conceptual metamodel

4.4. DEVELOP HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL FRAMEWORK :

Framework of developed conceptual metamodel for human centred software development (HCSD) shown in below figure 4.4.This developed metamodel has five different stages for develop software according to end-user's quality and requirements satisfaction point of view and before release the product we use the validation phase to

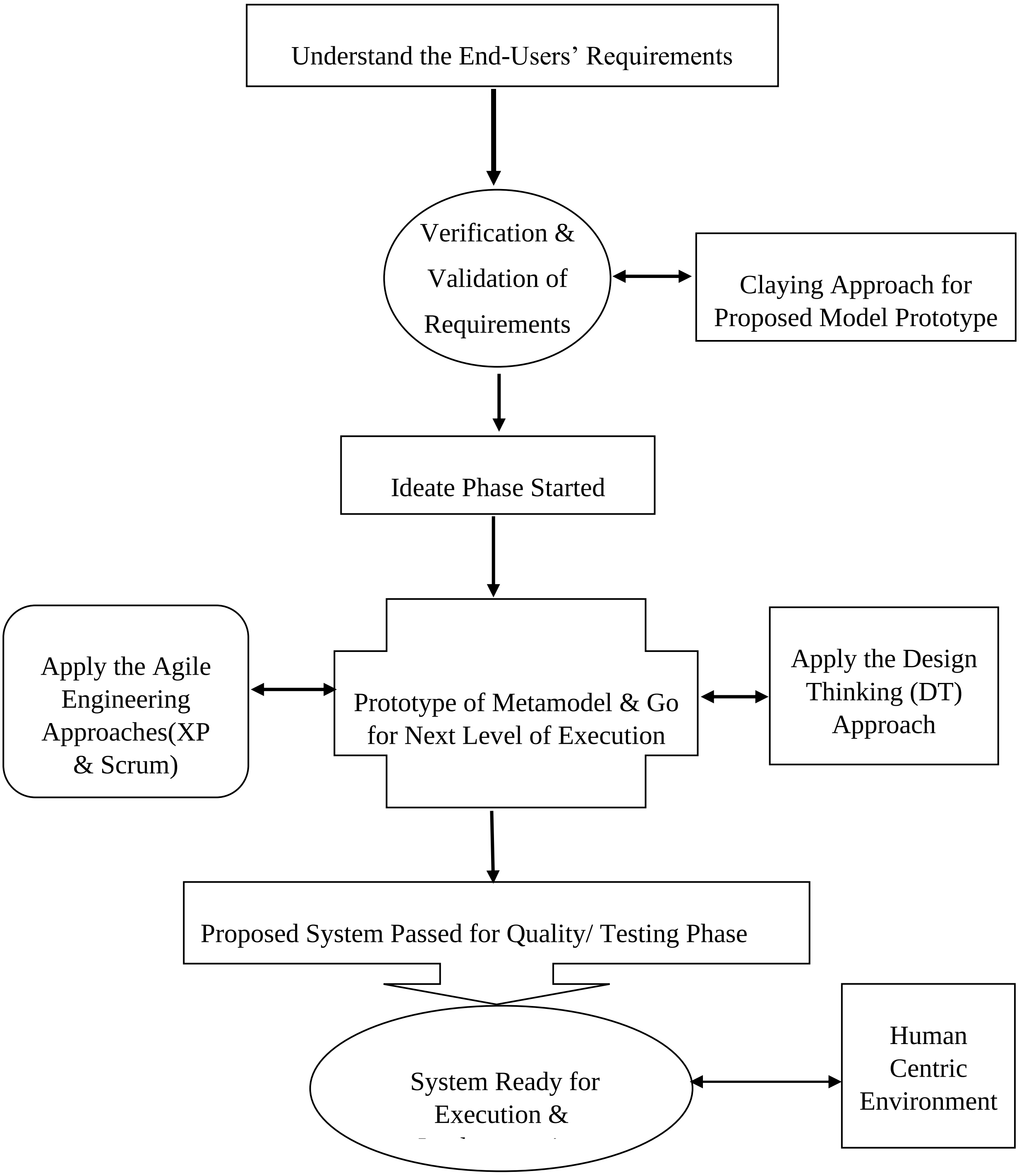

Understand the End-Users' Requirements
Verification & Validation of Requirements
Claying Approach for Proposed Model Prototype
Ideate Phase Started
Apply the Agile Engineering Approaches(XP & Scrum)
Prototype of Metamodel & Go for Next Level of Execution
Apply the Design Thinking (DT) Approach
Proposed System Passed for Quality/ Testing Phase
System Ready for Execution &
Human Centric Environment

Figure 4.4. Develop human centred software development (HCSD) conceptual metamodel framework [146-148]

enhance the quality of software/product and also increase the efficiency and usability of software designing aspects and reach the development on expected expectation level of stakeholders and users [149]. Develop human centred software development (HCSD) metamodel we not used any predefined and specify technique of agile engineering and design thinking (DT) approach for human software development but it is all depend on the requirements, designing environment and constraints and others feedback gets by users regarding the developmentation, so in this developed model we integrated "design thinking (DT) with agile engineering Scrum, Lean and XP" to get "user acceptance from the end result of product and services and also satisfy stakeholder point of view and business interest as well [150]. This framework core aspect is develop clay/prototype involved operational responses and feedback of end-users and product stakeholders to validate and quality of process designing phases.

Requirements Behavior Phase: In this, we "analyze or filters the non-technical and technical requirements and factors" on the basis of designing thinking (DT) and user's point of view and understanding capability and this phase we give name is "understand the human behavior phase also [151].This phase of human centred software (HCSD) firstly recolonized the requirements according to human centric approach.

User Requirements Validation Phase: This is called "verification and validation (VV) phase of requirement and feedback". In this we concentrate on input received by first phase and verify and validate the feedback or requirement through feasibility analysis approaches of agile engineering and design thinking (DT) for this purpose we involved the "agile XP and Scrum technique and parallel way used design thinking (DT) empathize approach" for decide the requirements according to client view and prospective and make view level for designing phase in this we kept end-user's feeling, expectations and thoughts form the software development or product designing also [152]. This phase directly release response and requirements after the verification and validation phase to next conceptually claying phase in where the system development prototype is built-up and decided and if there is any issue examined/arise here then requirements again pass to requirements verification and validation phase and after the validated the requirements then output pass to next phase of model.

Ideate Phase : This is called "technicality decided phase" in here we decided technology and methodology those apply on the "previous of requirement and feedback phase" and used the design thinking approach and agile engineering methodology: Scrum, Lean and XP encapsulated inherit methodology for provide the solution for human centric problem and requirements [153].

Conceptual Integrity and Prototype Phase: This is called “system designing phase” and designing aspects and different prospective of system design regarding end-users’ point of view and demands finalize here and for this purpose we used the “design thinking (DT) ideate approach”. In this phase actual prototype of designing is ready for implementation and further acceptance level of end-users [154].

Quality and Execution Phase: This is final stage of conceptual metamodel in this stage, system is become ready for the end-users’ input and requirements and validate the execution effectiveness and efficiency up to decided objective and pre-decided requirements those declared earlier phase of system designing for this purpose we used the design thinking (DT) test technique for quality assurance of developed system or software application and product [155].

4.5. ALGORITHM FOR DEVELOP HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL :

Alogorithm for develop human centred software development(HCSD) model discuss below and by the use of this algorithm we compute user acceptance index factors and performance index factors also by the derived methods/formulas.

- Input: User Requirements through involvement in design phase
- ***Step 1:*** Compute the project requirements of end-users.
- Project requirements= (S+F) * user acceptance ratio
- S=SRS requirements
- F= FRS requirements
- ***Step 2:*** Analyze the user feedback in middle of software development life cycle (SDLC).
- ***Step 3:*** Involved agile engineering XP and Scrum Sprint methods
- ***Step 4:*** Tested the agile engineering sprints methods according human centric requirements.
- ***Step 5:*** Agile engineering human centric result forward to validation phase.

- ***Step 6:*** Verification and validation of requirements specification done.
- ***Step 7:*** Ideate phase of process designing started.
- ***Step 8:*** Involved agile human-centric and design thinking (DT) technique.
- ***Step 9:*** Prototype of Human centred software development ready.
- ***Step 10:*** Developed system is ready for implementation & execution phase
- ***Step 11:*** Apply User acceptance index factors count for capture the real feedback by users.
- **Step 12:** Human-Centric environment for post feedback and responses by end-users
- Output: Human-Centric product ready for execuation and user acceptance.
- End.

The user acceptance index in human centred software development model (HCSD) calculated by given formulas:

Total efforts count during traditional SDLC= (software development efforts*time)/100

Total efforts count in HCSD =∑ (efforts by agile XP and Lean Programming + Design thinking factors)/100

Total user acceptance index count percentage in HCSD model is:

∑ (Efforts/Time) *100

Total time acceptance index count is:

∑ (User Satisfactions/No of Validation failure steps) *100

Note: efforts and time count taken at the integration level of agile XP, AM and Design Thinking (DT) to reduce total efforts during process development stages and increase user acceptance index factors.

4.6. Conclustion :

This phase of research, provides different features of design and develop human centred software development (HCSD) conceptual metamodel and their working environment and also discuss the architecture and framework of develop human centred software development (HCSD) conceptual metamodel. In the last of this chapter discuss the different phase of conceptual metamdel and the working of each phases of develop human centred software development (HCSD) conceptual metamodel to fullfill the objectives of the research as declared earlier stage of reseach work.

CHAPTER - 5
RESULT AND DICUSSSION

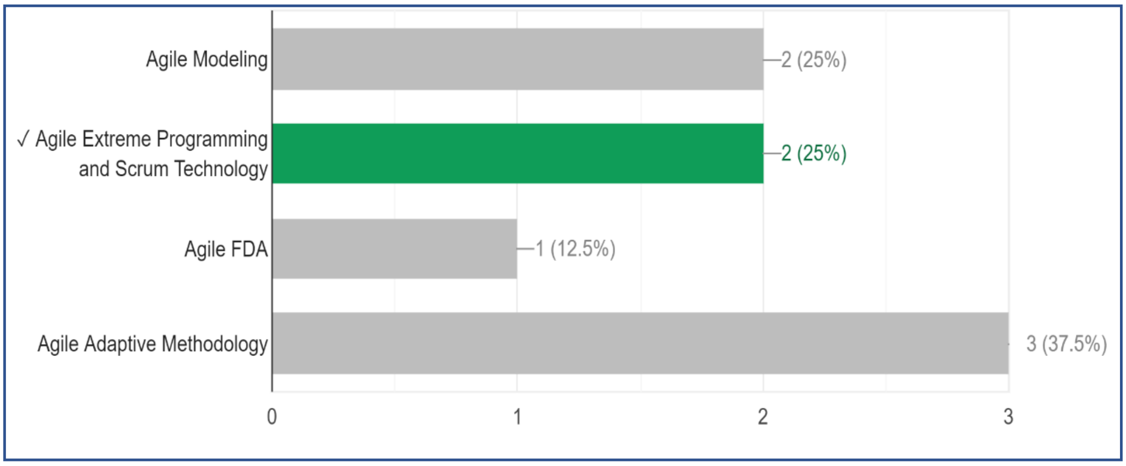
RESULT AND DISCUSSION

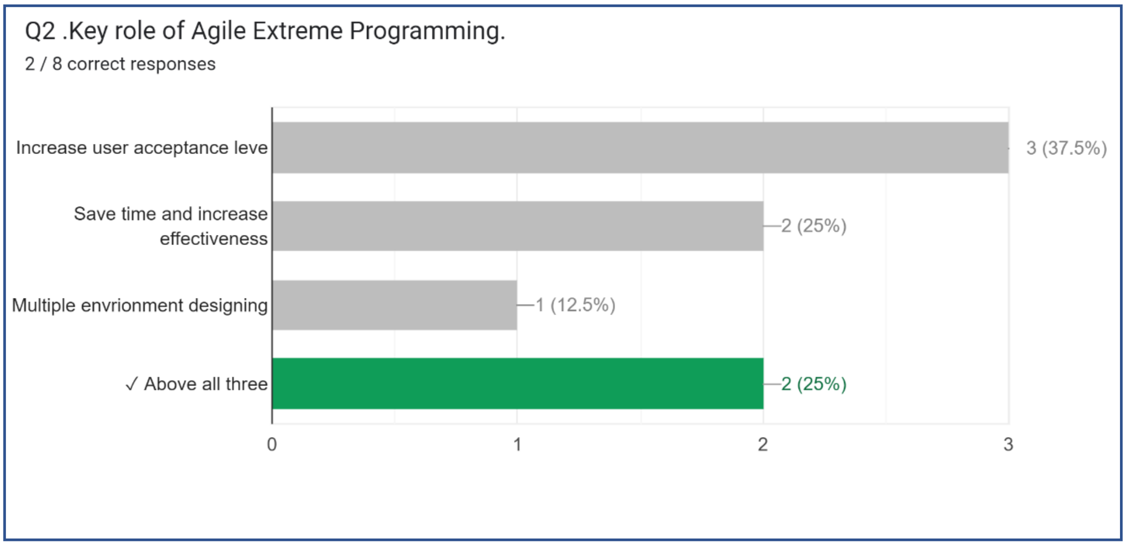
In this chapter result and analysis objectively and neutrally explain and present the qualitative analysis, fact, methodologies, framework architecture and performance analysis and provides statistics data and given relevant results and provides analytical fact and figure in systematically manner and in order to determine the objectives as decided in earlier stage of research work. This chapter provides comparative analysis of agile engineering approaches, quality index and maturity index, analytically examined and provides statistically performance comparison results between agile engineering approaches, design thinking (DT), and integrated designed and develop human centred software development (HCSD) model, this chapter also provide project quality index factors through human centred software development (HCSD) model and in last of this chapter critically examined result provides for the user acceptance index, cost and time index factors through integrated designed and develop human centred software development (HCSD) model. This chapter result had provided significant analytically results in respect of agile engineering, design thinking (DT) and integrated designed and developed human centred software development (HCSD) model based on agile engineering and design thinking (DT) approaches and in brief this designed and developed human centred software development (HCSD) model save the cost and time and increase the user acceptance level and also increase the reliability of software development.The result analysis based on survery questionaries and sample size is more than 50+ and release to software industry experts persons for their feeback , responses and afterward research analysis and dicusssion phases is completed and analytical data representation shown in this chapter.

Note: The result anlaysis fact and figure drived from analysis of survery,analysis feedback conducted in this reseach work and treated as data set value and for this result analysis survey based on serval questions and some of them sample questiojs set discusss below as Q1-Q5 and Q8 for data integrity and valitadion of reseach analysis fact and figure according to software industry parameters and standards.This is survery conducted during research and get their responses on agile-humanc cenrtric and design thinking integrated human centred software development (HCSD) conceptual metamodel.

Q1. Best approach of Agile Engineering for Human-Centric Development

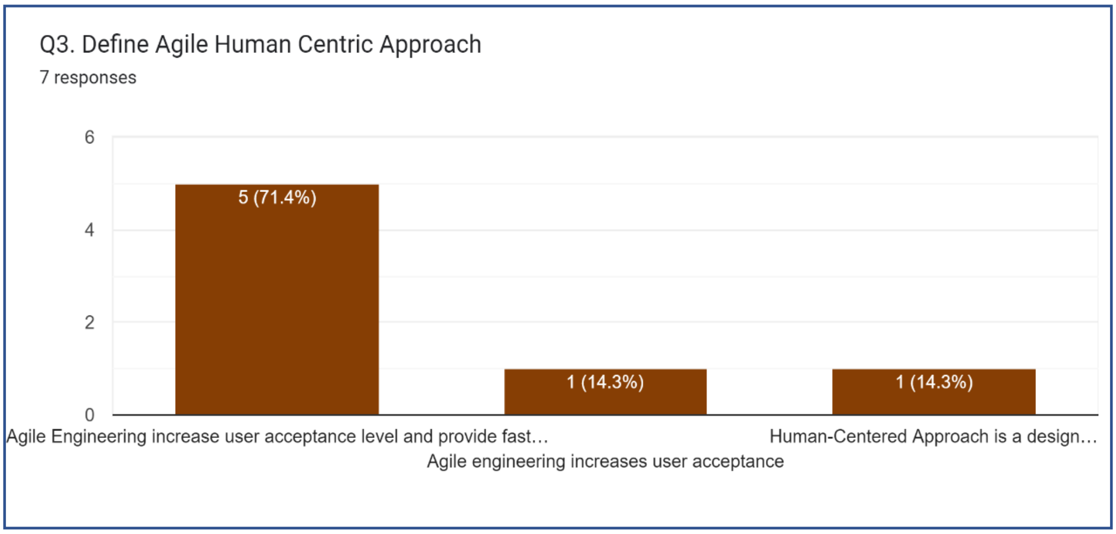
2 / 8 correct responses





Q2 .Key role of Agile Extreme Programming.

2 / 8 correct responses

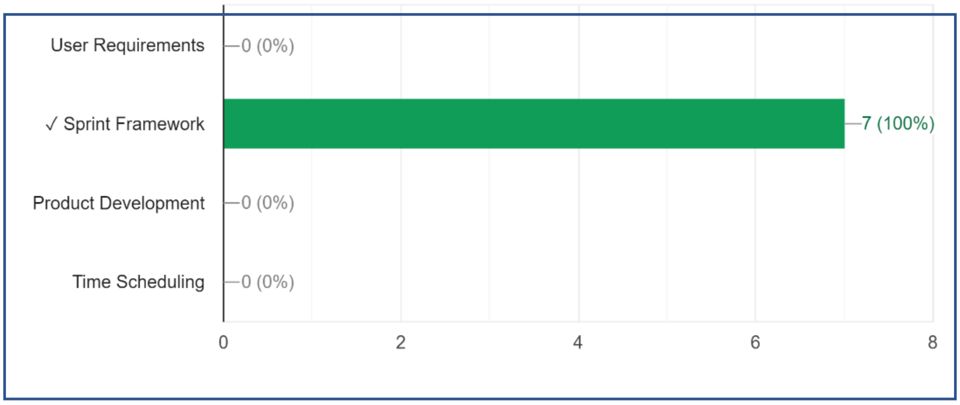


Q3. Define Agile Human Centric Approach

7 responses

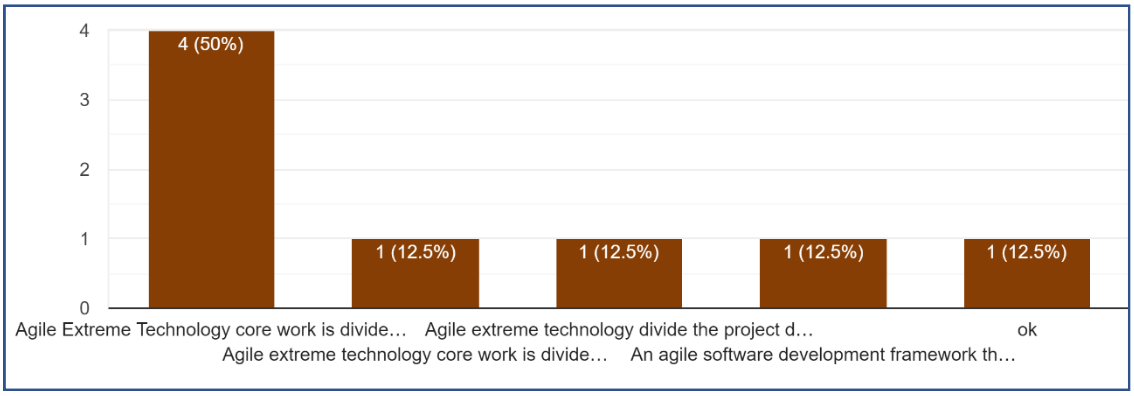
Q4. Agile Scrum Technology core working envrionment

7 / 7 correct responses



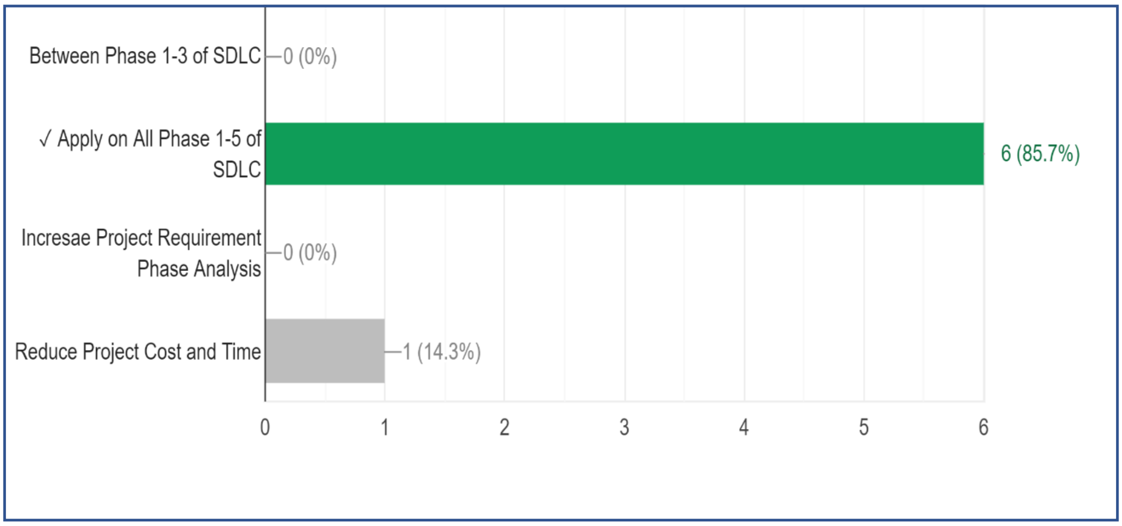
Q5. Agile Extreme Technology Core working Feature is.

8 responses



Q8.Propsed Conceptual Human Centred Software Development Model Increase User Feedback and Acceptance Level

6 / 7 correct responses



5.1. COMPERATIVE RESULT ANALYSIS OF AGILE ENGINEERING APPROACHES AND THEIR QUALITY IMPACT FACTORS :

In this phase anlyis of impact factors of agile engineering approaches in respect of quality and maturity user acceptance maturity index, for this result anlayis parameters are deriven from the taken data set values and responses. This result analysis phase comparatives analysis done on "six different parameters/attributes data values: agile engineering approaches, SDLC phases cover by agile engineering, core phase of SDLC cover, efforts (time/cost) respective, user acceptance level, and quality rank index facotros count". The result of impact factors analysis and performance analysis are shown in below table 5.1 and quality rank factors of different agile engineering approaches and level of maturity analysis also measured.

Table 5.1. User acceptance and quality impact factors analysis by agile engineering methodologies

S. No.	Agile Approaches Selection (Type)	SDLC Phases Cover in Agile Engineering	Level of SDLC Mainly Cover	Efforts (Time/Cost)	User Acceptance Level	Quality Rank (Out of 5 Scale Value)
1	Extreme Programming	Native principles and values	Phase 1-4	Middle	Given Constraints	>2<4.5
2	Agile Modeling (AM)	Address Complex Issue	Phase 1-3	Incremental Level	Focus on Group Level	>1<3.5
3	Scrum	Build a Backlog, Process	Phase 1-5	Average Level	Iteration over exhaustive	>3<4.25
4	Crystal Methodologies Family	Efficiency and Habitability	Phase -3 (tools and standard)	Crystal clear (Always Counted)	Project safety	>4<4.5
5	Feature-Driven Development	Client value feature view	Phase 1-3	Tangible	Timely and client view level	3<4
6	Adaptive Software Development	Based on rapid application development (RAD)	Phase 1-5	Not decided	Continuous	>4<5.0

Comparative analysis of different agile engineering approaches and user acceptance level retention factors are shown in below figure 5.1 and in this as we seen some approaches of agile engineering user accpetanc level index more than 70%.

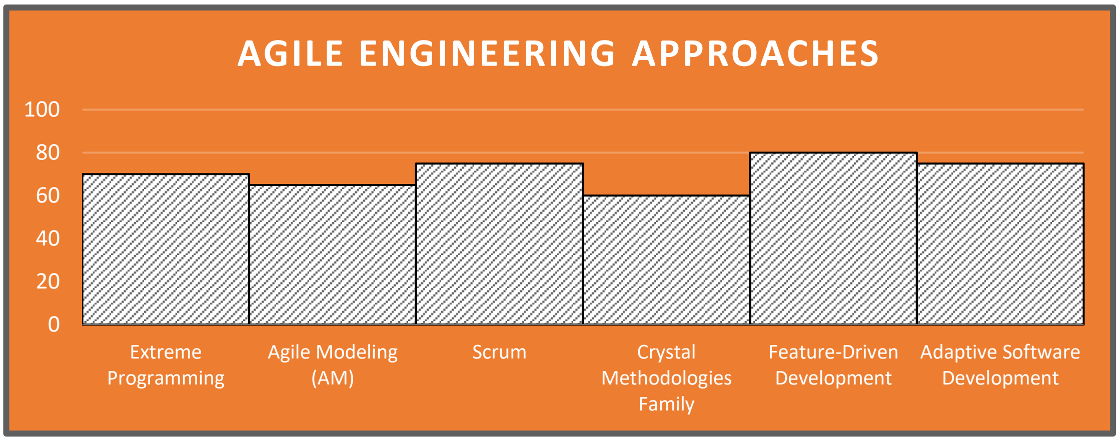


Figure 5.1. Comparative analysis of different agile engineering approaches and user acceptance levels

In this phase of result analysis data retrieval and apply data retentsion best practices for comparatives studies of different agile engineering approaches. Agile engineering approaches determine the acceptance level of project team in term of internal communication acceptance index and make it as far possible through the involvement of customers/client in strating phases of software development and in this section we critically analysed the different agile engineering approaches and their user acceptance level index through the derived data set of result analysis phase.

5.2. PROJECT QUALITY INDEX FACTORS ANALYSIS THROUGH AGILE ENGINEERING :

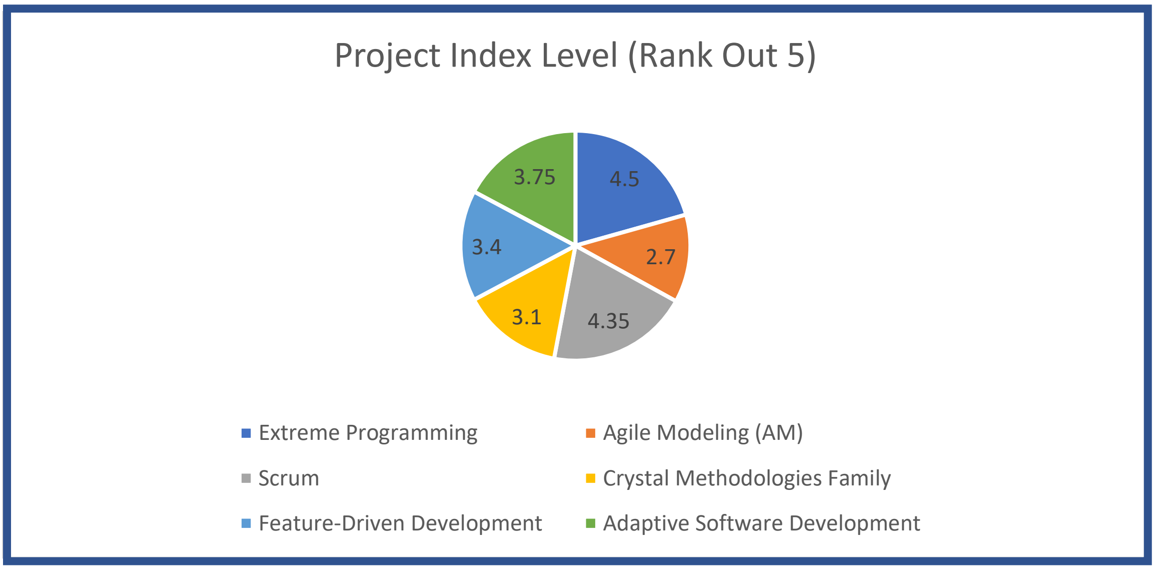


Figure 5.2. Project quality index factors analysis through agile engineering

Project quality index play significant role in respect of user reliability and quality point of view and as shown in above figure 5.2 project quality index (rank index out of 5) in respect of different agile engineering approaches as like extreme programming (XP), agile modeling (AM), scrum, crystal methodologies family, feature driven development and adaptive software development approaches. As we seen in this phase of result analysis that extreme programming (XP) and scrum approach quality index factors more measured >= 4 (out of rage of quality index 5), so this reason in the developed human centred software development (HCSD) conceptual meta model used agile engineering human-centric approaches extrem programming and scurm to ensure the end-users requirments and standards.This developed conceptual human centred software development (HCSD) project implementation rate and budget allocation vary in agile engineering approach to non-agile engineering approaches as shown in below figure 5.3 project acceptance level increase in agile engineering practices to non-agile engineering approaches.

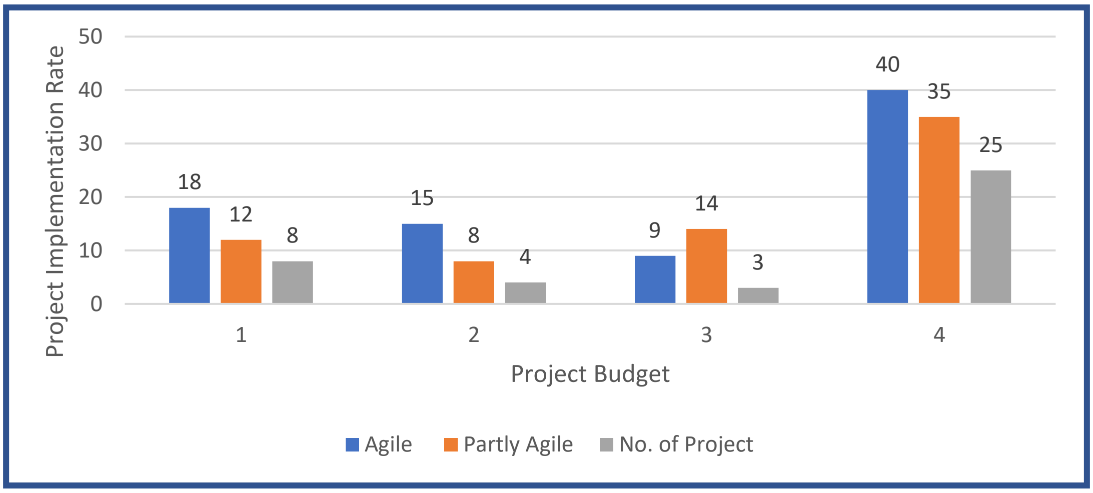


Figure 5.3. Project implementation rate and budget allocation analysis through agile engineering and non-agile engineering practices

5.3. PROJECT PERFORMANCE AND IMPACT FACTORS ANALYSIS THROUGH DESIGN THINKING (DT) :

Design thinking (DT) approach increase the capability of software development in human centric point of view In this session we done performance analysis and impact factors analysis of software development designing different phases through design thinking (DT) approach. In here we get role of design thinking (DT) phases: empathize, define, ideate, prototype and test in human centred software development (HCSD). Impact factors, user acceptance leve and quality index of project development and as stakeholders and end-users point of view play significant role and design thinking (DT) capture/meet all these goals in realistics manner and give possible realiable solutions operations level.In this result analysis phase performance analysis measured through some factors of software development life cycle (SDLC): user requirements analysis, designing schedule and activity started, system design ready stage, quality assurance phase. In this phase of result analysis and discusstion data set values/attributes derived from impact factors and user acceptance level and quality index comparatives analysis in respect of software development life cycle (SDLC) different phases through design thinking (DT) as given below table 5.2 shown the design thinking (DT) stages and project maturity level and quality rank scale value and also find out the user acceptance level of project.

Table 5.2. User acceptance levels and quality index factors analysis through design thinking (DT) approach

S. No.	Design Thinking (DT) Stages	SDLC Phases Cover in Agile Engineering	Level of SDLC Mainly Cover	Efforts (Time/Cost)	User Acceptance Level	Quality Rank (Out of 5 Scale Value)
1	Empathize	User requirements	Phase 1-2	Low	Very high	4-5
2	Define	Requirement designing	Phase1-3	Low	Accepted	3-4
3	Ideate	Designing schedule and activity started	Phase 2-3	Average level	Middle level	4-5
4	Prototype	System design ready	Phase 2-4	Very clear	Standard	5
5	Test	Quality assurance	Phase 4-5	User assurance	Achieved all predefined objectives	4

5.4. USER ACCEPTANCE DISTRIBUTION GRAPH OF DESIGN THINKING (DT):

In this session, we get that define, ideate, and prototype phases user acceptance index factors are at maximum level of stages through this designed and develop human centred software development (HCSD) model and get the user acceptance level as on validated stage of requirements specification and user point of view. By this developed process management metamodel we get that, “ideate phase of design thinking makes significant impact ratio as we seen in figure maximum value percentage is more than >90% of total acceptance level of user” in developed model. In this pahse of result analysis we count the user acceptance level through desing thinking (DT) approach.User acceptance index measured on empathize, define, ideate , prototype and test phases of design thinking (DT).Result analysis this phase used acceptance levels distribution graph for design thinking (DT) different stages.

Design thinking increased the user acceptance level scenario and also increase the project execution level at the user operational behavior manner and below figure 5.4 shown the design thinking (DT) stages and their different level of user acceptance impact factors distribution graph.

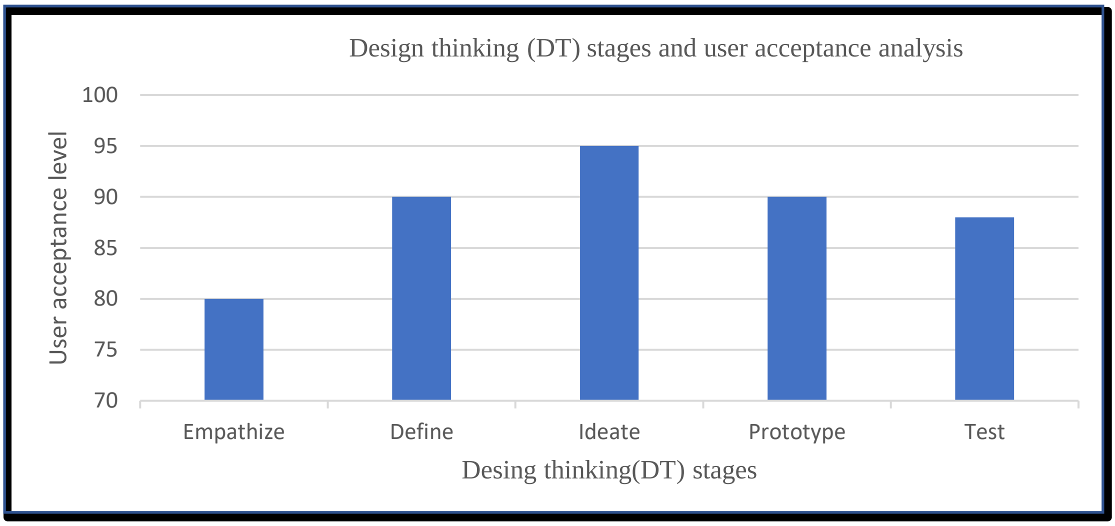


Figure: 5.4. User acceptance levels distribution graph of design thinking (DT)

5.5. PROJECT SUCCESS FACTORS ANALYSIS BY INTEGRATED DEVELOP HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL [165-167]:

Human centred software development (HCSD) conceptual metamodel reduces the time-complexity and increase the efficiency between the different stages of software development life cycle (SDLC) through output.In this phase we discuss the comparative analytical result of design and developed human centred software development (HCSD) model in respect of time duration to developed software the this developed model. In this research, we analyze design time, implementation time, maintenance time and code size by the used of human centred software development (HCSD) and validated the user acceptance with reference to time duration and this research we get the result that code impact gives most significant impact on human-centric process development. By this result, we get that code complexity play approx. 50% span of software designing and process management time and cost and developed conceptual process management mainly focus on user acceptance.

Project development analysis level through develop human centred software development (HCS) metamodel by the integration of agile and design thinking (DT) methodologies as shown in graphical manner in given below figure 5.5 and data set showing that time complexity reduces in every stages of software development life cycle (SDLC) phases and not reduces upto 50% in SDLC stages.

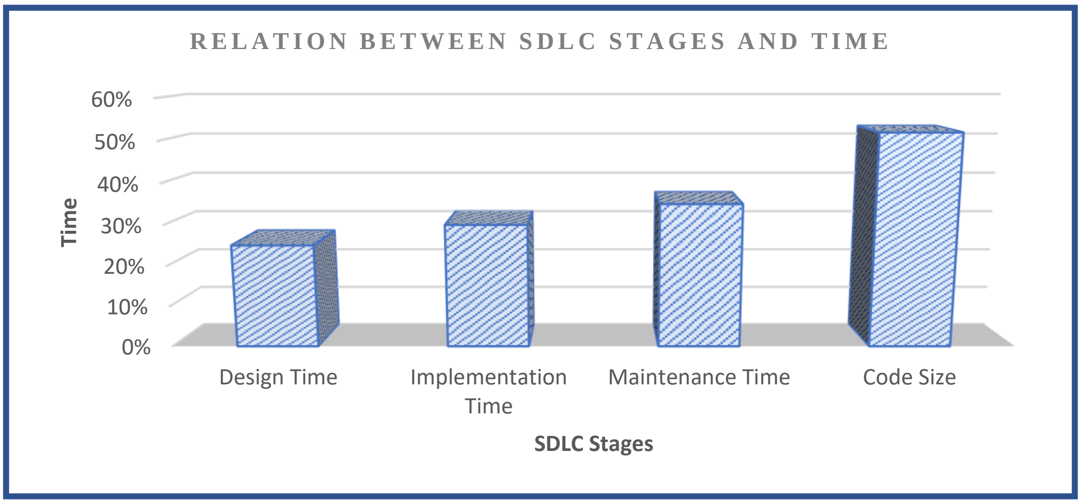


Figure:5.5. Project development analysis through develop human centred software development (HCSD) conceptual metamodel

Integrated design and develop human centred software development (HCSD) conceptual metamodel some success factors directily depends on user satisfaction and quality, and parallelly others index factors focuses on data modularity between project succss factors and reusable components like: responses and acceptance factors of project delivery.This session result analysis gives data interpretation that human centred software development (HCSD) metamodel increae user acceptance ratio as given below:

Specification > = User Acceptance (80% to 90%) based project size and complexity

Designing and Architectual > = User Acceptance (70% to 85%) based on project feasibililty and human-centric requirement speicications.

In this developed conceptual model, we analysis and get responses level and success factors of project, percentages of project success and user acceptance success rate as discuss in below table 5.3 and this data analysis we get to passed some parameters and factors like: user involvements, project management, designing, project testing and quality, and user acceptance factors and business stakeholder satisfaction level and objective fulfillment ratio and in last rate of project execution factors and level.

Table:5.3 Compute success factors of project executions, percentage of responses and user responses through agile engineering approaches

S.No.	Project Success Factors	Percentage of Responses	User Satisfaction
1.	User Involvement	15.9%	High
2.	Upper Level Management System	13.9%	Moderate
3.	Requirement Specification	13.0%	High
4.	Designing & Planning	13.0%	Moderate
5.	Execution Expectation	8.3%	Between Low to Moderate
6.	Achieve Project Milestones	7.8%	High
7.	Competent Team Members	7.0%	Low
8.	Ownership	5.3%	Average
9.	Vision & Objectives	2.9%	High
10.	Realistic Staff	2.4%	Average
11.	Other	13.9%	Moderate

5.6. COUNT PROJECT FAILURE FACTORS AND PERCENTAGE OF RESPONSES IN PROJECT DEVELOPMENT:

This phase of result analysis and discussion focus on compute failure factors of project execution in respect project success factors and user statifaction and acceptance index factors percentage and their overall data dependency in project development.In this phase of research, we count the project failure factors and percentage of responses get by end-users during software development through human centred software development

(HCSD) model by using metric transformation algorithm as discuss given below table 5.4. In this table we involved agile-human centric XP, and Scrum approach to count the project failure factors, and user responses factors and this result we mainly takes input the user behavior and factors that are not directly deal with end-user requirements and technology skills.

Table:5.4 Compute failure factors of project executions, percentage of responses and user responses through agile engineering approaches

S.No.	Project Success Factors	Percentage of Responses	User Satisfaction Scale (0-5)
1.	Undeclare Requirements	13.0%	0-1
2.	Users Involvement Less	12.5%	1-2
3.	Lack Resources Specifications	10.5%	0-1
4.	User Expectations	10.1%	0-2
5.	Lack of Expert Opinion	9.2%	1-2
6.	Changing Requirements	8.8%	Not Focus
7.	Lack of Planning	8.0%	Non-Realistic
8.	Not Need Longer	7.0%	1-2
9.	Lack of Management	6.7%	1-2
10.	Technology Skills Knowledge	4.7%	Not Focus
11.	Other	9.5%	0-2

5.7. SOFTWARE PROJECT PERFORMANCE ANALYIS BY THE USAGE OF AGILE ENGINEERING, DESIGN THINKING (DT) AND DEVELOP HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL:

In this session of research analysis we shown the performance analysis comparison factors of software development life cycle (SDLC) and their impact factors through agile engineering, design-thinking (DT) and also point out the human centred software development (HCSD) total factors count in respect of cost, design acceptance, development time span and user acceptance of software development and provide analytical result on these factors of project development and acceptance levels and also gives the analytical result analysis on these factors. The sum of agile engineering, design thinking (DT) and develop conceptual human centred software development(HCSD) model shown in below table 5.5 and by this result analysis, we get the "sum of performance analysis factors maximum through developed human centred software development ratio of sum is more than 2.8/4". As we get analytical data that user acceptance measured maximum satisfaction index through HCSD and their performance analysis graphical representation shown in below figure 5.6 in respect of agile engineering, design thinking (DT) and develop conceptual human centred software development (HCSD) metamodel.

Table:5.5 Performance factors analysis through agile engineering , design thinking (DT) and HCSD approaches

Row Labels	Sum of Agile Tngineering	Sum of Design Thinking	Sum of HCSD
Cost	0.8	0.7	0.6
Design Acceptance Level	0.55	0.6	0.8
Development Time Span	0.85	0.7	0.5
User Acceptance	0.7	0.75	0.9
Grand Total	**2.9**	**2.75**	**2.8**

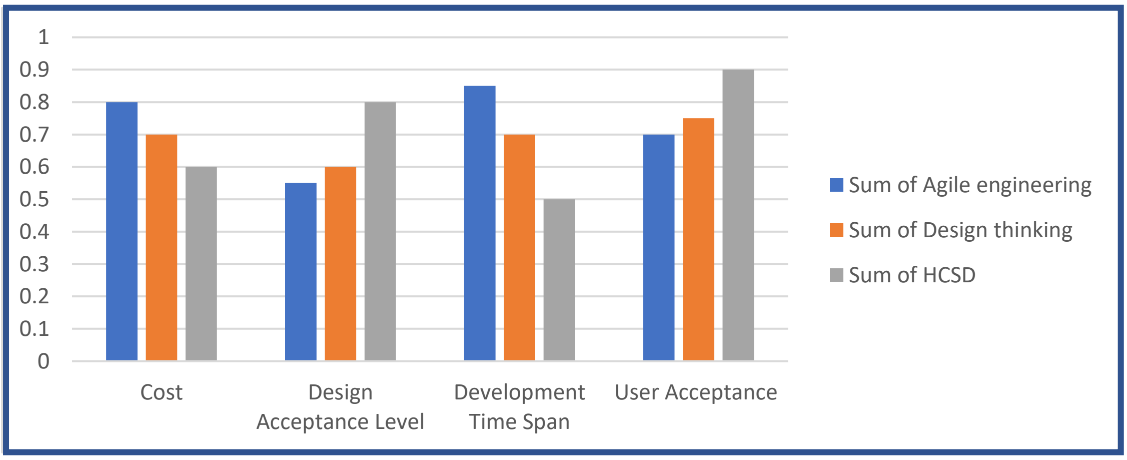


Figure:5.6. Graphical representation of SDLC factors and sum of agile engineering, design thinking (DT) and HCSD approach

5.8. ANALYSIS OF PROJECT DEVELOPMENT QUALITY FACTORS THROUGH DEVELOP HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL:

In this session of research analysis we emphasize and provide the project development index and quality assurance factors and their impact on developed human centred software development (HCSD) and in here we get the use of "developed conceptual process model we get index factors more than 95% (4.5 to 5) in all phases of software development phases" as shown in given below figure 5.7 diffeent phases of human centred software development (HCSD) metamodel and project exuecution index factors. As we seen through the data analysis that maximum user acceptance index factor possible through human centric requirements understanding phase and through the involvement of agile engineering conceptual techniques to relese the human centred software development (HCSD) process management and development.

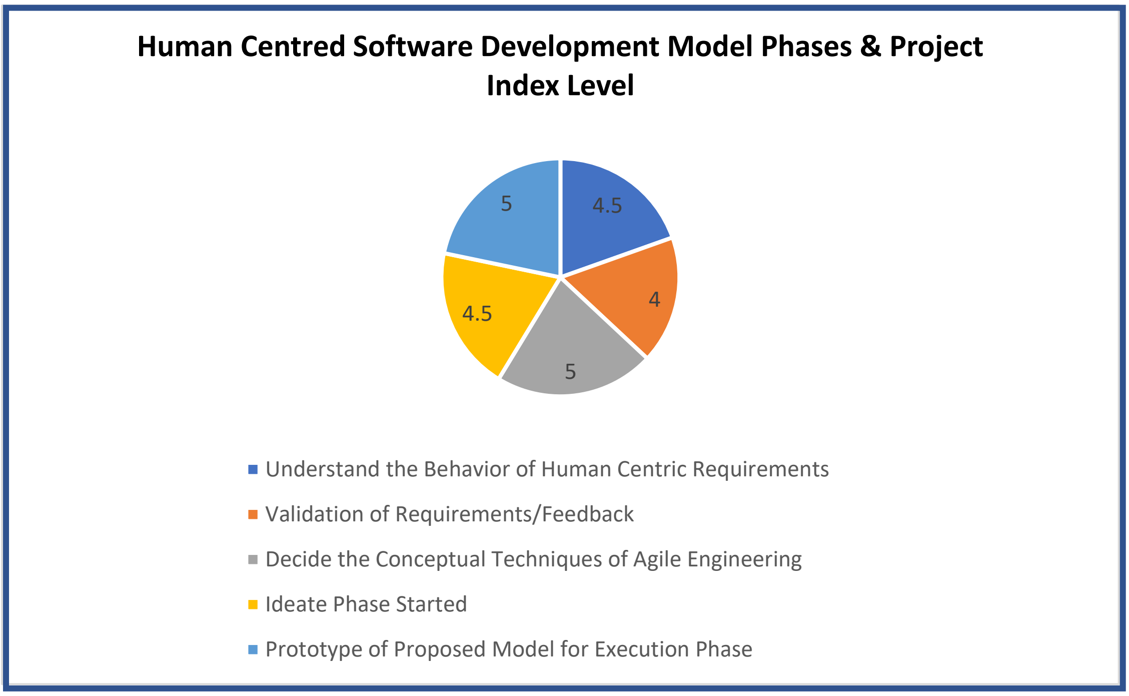


Figure:5.7. Develop Human centred software development (HCSD) metamodel development phases and project quality index factors

5.9. ANALYSIS OF RELIABILITY AND QUALITY INDEX OF SOFTWARE PROJECT BY DEVELOP HUMAN CENTRED SOFTWARE DEVELOPMENT (HCSD) CONCEPTUAL METAMODEL :

In this session of research analysis, we get reliability quality index ratio factors of software development phases by the developed human centred software development (HCSD) model for data retention rate, verification and validation factors count in respect of technology scenario and dependency and here we also get the analytical result that is human centric requirements stisfaction ratio is more than 20% and by the use this model we get ratio of execution phase is more than 23% as shown in figure 5.8. In here get the result that developed software process management metamodel mainly focus on validation of requirements and feedback of end-users and business stakeholder. Develop conceptual integrated human centred software development (HCSD) model maximum quality index value of project validation and quality assurance more than 37% of total reliability quality index. Reliability quality index count of project through comparative analysis by developed human centred centred software development (HCSD) metamodel that project acceptance level high when data behavioural values is maximum, between 14% to 23%.

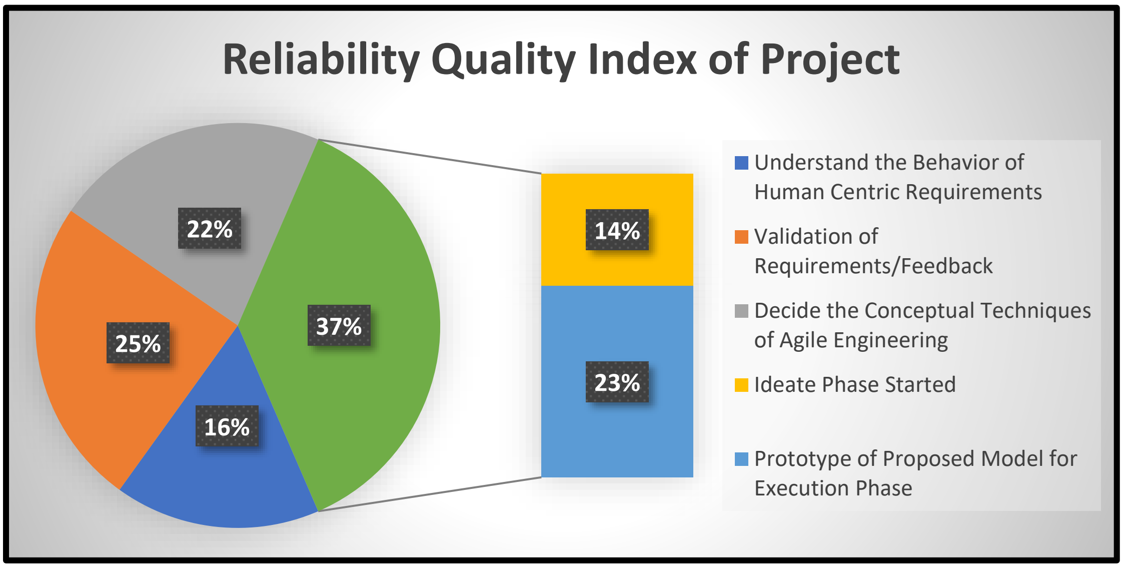
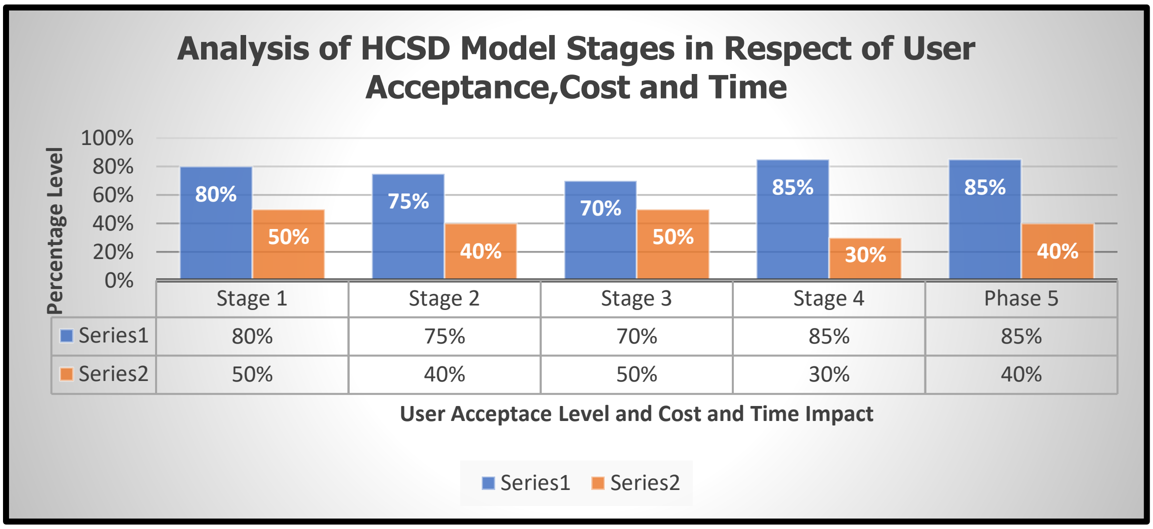


Figure:5.8. Reliability and quality count ratio index factors of project development phases by develop HCSD conceptual metamodel [166-167]

Graphical analysis representation of HCSD model different stages in respect of user acceptance, cost and time factors and index ratio comparison factors as shown in below figure 5.9 and also gives view on user acceptance level,cost and time impact in respect of 5 different stages of process development levels.



	Stage 1	Stage 2	Stage 3	Stage 4	Phase 5
Series1	80%	75%	70%	85%	85%
Series2	50%	40%	50%	30%	40%

Figure:5.9 Graphical representation of HCSD conceptual metamodel stages, user acceptance, cost and time index factors

5.10. CONCLUSION :

In this session of research analysis, we discussing develop human centred software development (HCSD) conceptual metamodel result analysis core data retrieval facts and results outcomes as listed below:

- Understanding and requirement behavior of human-centric specification = **37%**
- Validation phase of technology specification = **25%**
- Conceptual techniques and ideate phase specification = **22%**
- Prototype and execution specification phase specification = **16%**
- Acceptance level in respect of cost is reduce more that **40% to 60%** by the used HCSD model (Source is figure 5.9)
- End-user acceptance level and feedback satisfaction index factor count is between **75% to 85%** (Series-1 figure 5.9)
- Time acceptance index factor count is reduced more **50% to 60%** of total time count through developed HCSD (Series 2 figure 5.9)

CHAPTER -6

SUMMARY

&

CONCLUSION

6.1. SUMMARY :

The Research work improves the software development designing and process management performance and effectiveness employing agile-human centric and design thinking (DT) approaches. Agile-human centric software development approach XP (extreme Programming) and Scrum technology work on human-centred software development and process management and other side design thinking (DT) approach involved in earlier phases of software development to develop software product according to end-user satisfaction and point of view. In this research we design and develop conceptual integrated process management metamodel using agile-human centric and design thinking (DT) approaches to release the graphical representation of software process and their development stages progress in the middle and running stages of software productivity. Agile engineering techniques increase efficiency and effectiveness in software designing and develop software in different phases and sprints in parallelly to minimal uses of resources and gives optimization solution on the user-requirements. Design thinking (DT) approach uses five stages of software designing and development: empathize, define, ideate, prototype, and test for taking user feedback and responses to decided designing phases and mindset of software development. In this research we design and developed human centred software development (HCSD) model for human centred software designing and process management.

Chapter 3 discussed the research methodologies to design human centred software development (HCSD) metamodel using the integration of agile-human centric technologies XP and Scrum and design thinking (DT) approaches in conceptual manner of developmentation of this metamodel.

Chapter 4 discussed the framework and different stages of design and develop conceptual integrated metamodel called human centred software development (HCSD) model. This metamodel has mainly 6 stages of software development: verification and validation phases of requirements, ideate phase of designing, integration phase of agile-human centric and design thinking (DT) approaches, prototype phase, execution & implementation phase of system, and last phase is human-centric environment of user uses and operational implementation and feedback and responses for business stakeholder satisfaction and objective fulfillment. In section 4.4.1 we discussed algorithm of Develop Human Centred Software Development Model (HCSD).

Chapter 5 discussed the result and analysis and here we analyzed that, conceptual metamodel increases user acceptance index level validity between 85 % to 95% in designing and ideate phases by the uses of human centred software development (HCSD) model and this consistency of user acceptance index factors is always on higher stages in all the phases of software designing and process management in this design and develop human centred software development (HCSD) conceptual integrated metamodel.

6.2. CONCLUSION :

The results analysis shows that most of the integrated models are used throughout the software life cycle (SDLC), noting that the software development model that uses the functionality of design thinking (DT) approach released by the International Organization for Standardization (ISO) and Scrum technique as an agile methodology optimize the process development and provides the most frequently used integrated software development model based on these approaches. By this research the design and developed integrated human centred software development (HCSD) conceptual metamodel resulted in a greater effectiveness, approximation of end users and the designing and development team, improving the development quality and usability of the software. This study aims to evaluate how the agile engineering software development (AESD) approach along with the design thinking (DT) are integrated and possible the human centred software development (HCSD) practices and environment more realistic, and feasible manner accordingly software engineering designing parameters and quality standards.In this research critically reviewed and verified the agile software development (ASD) engineering methodology and simultaneously applied design thinking (DT) approach in integrated manner for a human centred software development (HCSD).The main advantage of this "conceptual human centred software development metamodel are metrics suite, reduced the development time, cost and efforts along with most important feature is reusability of agile and design thinking (DT) component in very easy level and in order to realize the reuse of software component effectively to fulfill human centric component based software application in standard and quality manner. This design and develop conceptual metamodel increase user acceptance index factors in all software development phases 4> out of indexting ranking count 5 and also accelerate the efficiency and effectiveness of end user operationals statisfactions and quality assurance.

6.3. FUTURE SCOPE:

In future, this design and develop human centred software development (HCSD) conceptual metamodel of software development will be accepted and implemented in software development industry and gets their feedbacks/responses and also capture their issues and bugs and then re-designed the software requirement according to the human centric need and satisfaction.The getting real time operational responses and requirements behavior of user end this process management metamodel become more open and quality oriented and deal with higher level of satisfaction and archived user point view more admirable and validated way in future point of view.In future this develop human centred software development (HCSD) concpetula metamodel will be becomer more realistics and portable approach by the operational end feedbacks by the ensure users scenariors and requirements behaviors.

CHAPTER -7

REFERENCES

REFERENCES

1. Rajeev Sharma, & J.N.Singh, "Human Centred Software Development Approaches",3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), 17-18 Dec. 2021.

2. Rajeev sharma, J.N. Singh, "A Critical Review of Surveys Emphasizing on Agile software engineering", solid-state journal vol 64.2, 2021.

3. Rajeev Sharma[1], Jitendra Nath Singh[2] , "Design-Thinking and User's Requirement Engineering: Human Centred Development",International Journal of Mechanical Engineering, 1,2 Galgotias University, Greater Noida, Uttar Pradesh, India, ISSN: 0974-5823,Copyrights @Kalahari Journals, Vol.6, Special Issue, Nov-Dec. 2021.

4. Manoj Palsodkar, Gunjan Yadav and Madhukar R. Nagre, "Recent trends in agile new product development a systematic review agenda for future research, Benchmarking" An International Jounal, ISSN :1463 -5771, Volume 30 Issue 9, Published, 8 September 2022.

5. Julio Cesar Pereiraa* , Rosaria de F. S. and M. Russoa a Uninove "Design Thinking Integrated in Agile Software Development" A Systematic Literature Review- International Conference on Health and Social Care Information Systems and Technologies, CENTERIS/ProjMAN/HCist 2018.

6. Stefan Sauer, "Human – Centered Software Engineering for Changing Context of Use", IFIP WG 13.2 Workshop at Interact, Aug 27, 2021.

7. Luis Corral[1] and Ilenia Fronza[2], "Design Thinking and Agile Practices for Software Engineering: An Opportunity for Innovation", [1]Monterrey Institute of Technology and Higher Education Queretaro, Mexico lrcorralv@itesm.mx, [2]Free University of Bozen-Bolzano Bolzano, Italy, ilenia.fronza@unibz.it Conference Paper,DOI: 10.1145/3241815.3241864, SIGITE'18, October 3-6, 2018, Fort Lauderdale, FL, USA. September – 2018.

8. Marzia Mortati, Stefano Magistretti, Cabirio Cautela, Claudio Dell'Era, "Data in design: How big data and thick data inform design thinking projects", Journal: Technovation Volume 122, April 2023.

9. Adel Alshamrani1 and Abdullah Bahattab2 1 Ira A. "Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model", Fulton Schools of Engineering at Arizona State University Tempe, AZ, 85281, USA 1 King Abdul Aziz University, Faculty of Computing and Information

Technology-North Branch, Jeddah, 22245, KSA 2 College of Telecommunications and Electronics (CTE), Jeddah, 21533, KSA,IJCSIA, International Journal of Computer Science Issues, Volume 12, Issue 1, No 1, January 2015.

10. 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, pp. 94 – 101, September 2010.

11. Ashwini Majumdar, Gayatri Masiwal, P.M. Chawan, “Analysis of Various Software Process Models” International Journal of Engineering Research and Applications, Vol. 2, No. 3, 2012.

12. N. Munassar and A. Govardhan, “A Comparison Between Five Models Of Software Engineering”, IJCSI International Journal of Computer Science Issues, vol. 7, no. 5, 2010.

13. Thorsten Schoormann, Maren Stadtlander, Ralf Kanckstedt, Act and Reflect: Integrating Reflection into Design Thinking, Journal of Management Information System, 02 Jan 2023.

14. Jim Hurst, “Comparing Software Development Life Cycles,” SANNS Software Security, 2014.

15. Sanjana Taya and Shaveta Gupta, “Comparative Analysis of Software Development Life Cycle Models,” IJCST Vol. 2, Iss ue 4, Oct. - Dec. 2011.

16. Jiujiu Yu “Research Process on Software Development Model”: IOP Conf. Series: Materials Science and Engineering 394 032045 doi:10.1088/1757-899X/394/3/032045, 2018.

17. E.J. Christie, Daniel Jensen, R.T. Buckley, D.A. Menefee “Prototyping Strategies: Literature Review and Identification of Critical Variables”,Conference: ASEE Annual Conference, June 2012, DOI:10.18260/1-2--21848.

18. Khansaa Azeez and Obayes Al-Husseini “Usage of prototyping in software testing”,[1] Babylon Technical Institute, Al-Furat Al-Awsat Technical University,51015 Babylon, Iraq,[1] Khansaa_aziz@yahoo.com [2] Ali Hamzah Obaid Babylon Technical Institute, Al-Furat Al-Awsat Technical University,51015 Babylon, Iraq. [2] alimk_iq@yahoo.com, Multi-Knowledge Electronic Comprehensive Journal for Education and Science Publications (MECSJ), ISSUE (14), Nov- 2018.

19. Raunaque Quaiser and Shivendra Kumar Pandey “Design thinking enabling innovation: A literature review”, Indian Institute of Management, Rohtak, Innovation the European Journal of Social Science Research 36(1):1-23, July 2023.

20. R.M. Zhang, D. Yang and J. Li, “Design of requirements negotiation tool based on WinWin theory”, Journal of Computer Engineering and Applications, issue 1, pp.100-104, 2009.

21. Michael Negnevitsky, “Artificial Intelligence: A Guide to Intelligent Systems” (Third Version), China Machine Press, 2012.

22. J.J.Yu, “A Short Course in Software Engineering”, Tsinghua University Press, 2015.

23. Z. Luo, S. Q.Yuan, J. L.Yuan and L. Li, “Software Engineering”, Posts &Telecom Press, 2017.

24. X.Zou, “Method of Construction: Modern Software Engineering,” Posts & Telecom Press, 2018.

25. Y.M. Du, S.X. Li, “Estimation Process Model for RUP Project”, Journal of Computer Science, vol. 40, issue 6, pp.21-26, 2013.

26. J.X. Xia, Z. Liu, X.B. Liu, Y.Song and J.J.Yuan, “Incremental Story Iteration Model Based on Rapid Application Development”, Journal of University of Shanghai For Science and Technology, issue 6, pp.578-583, 2014.

27. Mrs. Gunjan Behl and Mr. Nripesh Kumar Nrip, “A Study of Agile Software Development Model (Values, Principles and Characteristics)”, Assistant Professor, BVU Institute of management, Kolhapur. E-Mail – mailtogunjan@yahoo.co.in, Assistant Professor, BVU Institute of management, Kolhapur, nripesh.nrip1985@gamil.com, International Conference on Current Trends & Challenge, in Management, Engineering, Computer Application & Technology 2012.

28. Kallaya Tantiyaswasdikul, “Design Thinking for Innovation in Sustainable Built Environments and the Integration of an Inclusive Foresight and Design Thinking Framework”, International Journal of Sustainable Development and Planning,31 March 2023.

29. Torgeir Dingsøyr, Sridhar Nerur,Venu Gopal Balijepally, and Nils Brede Moea, “A decade of agile methodologies: Towards explaining agile software development explaining agile software development”, Jouranl of systems and software, Volume 85,Issue 6,June 2012, Pages 1213-1221.

30. Alves, R., Jardim Nunes, N, “Towards a taxonomy of service design methods and tools”, In: Falcão e Cunha, J., Snene, M., Nóvoa, H. (eds.) IESS 2013. LNBIP, vol. 143, pp. 215–229.Springer, Heidelberg 2013.

31. Abram D. Anders, “Human – Centered Leadership Development: A Communication – Based Approach for Promoting Authentic and Transformational Leadership”, International Journal of Business Communication, Nov-2021.

32. M.Broy, “Rethinking Nonfunctional Software Requirement Computer”, vol.48, no.5, pp.96-99, May 2015.

33. Tilley Scott, “Documenting software systems with views VI: lessons learned from 15 Years of research & practice”, Proceedings of the 27th ACM International Conference on Design of Communication. SIGDOC 09, New York, pp 239– 244. https://doi.org/10.1145/1621995.1622043.

34. Edward Park, “Human-centered Software Development, build stronger software by designing for human interaction”,Senior Software Engineer, May 28, 2021.

35. Didem Gurdur Broo, Okay Kaynak and Sadiq M.Sait, “Rethinking engineering education at the age of industry 5.0”, Journal of Industrial Information Integration, Volume 25, January 2022.

36. Paulo Sérgio Medeiros dos Santos, Alessandro Caetano Beltrão, Bruno Pedraça de Souza & Guilherme Horta Travassos, “On the benefits and challenges of using kanban in software engineering: a structured synthesis study”, Journal of Software Engineering Research and Development 6,Article Number 13, 2018.

37. Luis F. Mendivelso, Kelly Garcés & Rubby Casallas, “Metric-centered and technology-independent architectural views for software comprehension”, Journal of Software Engineering Research and Development 6,Article Number 16, 2018.

38. Conboy K, “Agility from first principles: reconstructing the concept of agility in information systems development”, Inf Syst Res 20:329–354, 2009.

39. Minelli R, Mocci A, Lanza M, Kobayashi T, “Quantifying program comprehension with interaction data” 14th International Conference on Quality Software, IEEE, USA, p 276–285, 2014.

40. Kaisa Savolainen, “User – Centred Design without Involving User: A Longitudinal Case Study in a Human- Centred- Design – Mature Company”, The Design Journal, Volume 24, issue -6, 2021.

41. Kevin Lano, Shekoufeh Kolahdouz-Rahimi, Javier Troya & Hessan Alfraihi, “Introduction to the theme section on Agile model-driven engineering, Software and Systems Modeling”, Volume 21, pages 1465-1467, 2022.

42. Andriole SJ, “The death of big software. Commun” ACM 60:29–32. https://doi.org/10.1145/3152722, 2017.

43. Anderson DJ Kanban, “successful evolutionary change for your technology business’, 3.8.2010. Blue Hole Press, Sequim, 2010.

44. Corona E and Pani F “A review of lean-Kanban approaches in the software development”, WSEAS Trans Inf Sci Appl 10:1–13, 2013.

45. Santos PSM and Travassos GH, “On the representation and aggregation of evidence in software engineering: a theory and belief-based perspective”, Electron Notes Theor Compute Sci 292:95–118, 2013.

46. Matković, P., & Tumbas, P, ‘A Comparative Overview of the Evolution of Software Development Models”, International Journal of Industrial Engineering and Management (IJIEM), 1, 163-172, 2010.

47. Wright and G. P, “Success rates by software development methodology in information technology project management: A quantitative analysis”, UMI Number: 3590342, UMI Dissertation Publishing, Pro Quest LLC, Michigan 2013.

48. Pernilla Agren, Eli Knoph, & Richard Berntsson Svensson, “Agile Software development one year into the COVID-19 pandemic”, Empirical Software Engineering, Volume 27, article number 121, June 2022.

49. David Itzik and Gelbard Roy, “Does agile methodology fit all characteristics of software projects? Review and Analysis”, Emerging Software Engineering (Springer), , Volume 28, article number 105, 14 July 2023.

50. Dyba T., & Dingsoyr, T, “Empirical studies of agile software development: A systematic review”, Information and software technology, 50(9), 833-859, 2008.

51. Sarah Laoyan: What is Agile methodology? (A beginner’s Guide), www.asana.com, _ October 2023.

52. Nerur, S., & Balijepally, V.Theoretical reflections on agile development methodologies. Communications of the ACM, 50(3), 79-83, 2007.

53. Strode and D. E, “Agile methods: a comparative analysis”, In Proceedings of the 19th annual conference of the national advisory committee on computing qualifications, NACCQ (Vol. 6), 257-264, 2006.

54. Abrahamsson, P., Oza, N., & Siponen, M. T, “Agile Software Development Methods: A Comparative Review1”, In Agile Software Development (pp. 31-59). Springer Berlin Heidelberg, 2010.

55. Qumer, A., & Henderson-Sellers, B, “An evaluation of the degree of agility in six agile methods and its applicability for method engineering”, Information and software technology, 50(4), 280-295, 2008.

56. Martin, A., Biddle, R., & Noble, J, “XP customer practices: A grounded theory”, In Agile Conference, IEEE, Aug, 2009 (pp. 33-40).

57. Siakas, K. V., & Siakas, E, “The agile professional culture: A source of agile quality”, Software Process: Improvement and Practice, Willey, 12(6), 597-610, 2007.

58. Iivari, J., & Huisman, M, “The relationship between organizational culture and the deployment of systems development methodologies”, MIS Quarterly, 31(1), 35-58, 2007.

59. Iivari, J., & Iivari, “The relationship between organizational culture and the deployment of agile methods”, Information and Software Technology, 53(5), 509-520, 2011.

60. Chow, T., & Cao, D. B, “A survey study of critical success factors in agile software projects” Journal of Systems and Software, 81(6), 961-971, 2008.

61. Aik _Ling Tan, “Design Thinking from Multiple Perspectives, Research in Integrated”, STEM Education, BRILL Online Publication Date: 07 June 2023.

62. Alicia Raeburn, “Extreme Programming (XP) gets results,but is it right for you?”, https://asana.com/resources/extreme-programming-xp, November 28th,2022.

63. Edmondson, A. C., & McManus, S. E, “Methodological fit in management field research”, Academy of management review, 32(4), 1246-1264, 2007.

64. Yin, R. K, “Case study research: Design and methods”, Sage publications, 2013.

65. Mazzur, G, “History of QFD – Introduction” http://qfdeurope.com/en/history-of-qfd/ (2015). Accessed 14 June 2019.

66. Sriram Rajagopalan Saji K Mathew, “Choice of Agile Methodologies in Software Development: A Vendor Perspective”, Department of Management Studies, Indian Institute of Technology Madras, India, Journal of International Technology and Information Management, Volume 25, Issue 1, Arctilce-3, 2016.

67. B.A. Kitchenham and S. Charters, “Procedures for Performing Systematic Literature Review in Software Engineering,” EBSE Technical Report version 2.3, EBSE-2007-01, Software Eng. Group.

68. Adel Alshamrani1 and Abdullah Bahattab, “A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model”, IJCSI International Journal of Computer Science Issues, Volume 12, Issue 1, No 1, ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784,2015.

69. Mihai Liviu DESPA, “Comparative study on software development methodologies”, Database Systems Journal vol. V, no. 3/2014.

70. Karthikeyan Chandran and Madhuchhandan Das Anudhe, “Agile or Waterfall development”,The Clementon Company dilemma, Journal of Information Technology Teaching Cases, Volume 12, Issue 1, January 9, 2021.

71. A. Spangler, “Cleanroom software engineering-plan your work and work your plan in small increments”, IEEE Potentials, vol.15, no. 4, pg. 29 – 32, 1996, doi: 10.1109/45.539962

72. M. Soeken, R. Wille, R. Drechsler, “Assisted behavior driven development, using natural language processing”, Proceedings of the 50th International Conference on Objects, Models, Components, Patterns, TOOLS 2012, 29- 31 May 2012, Prague, Czech Republic, Publisher: Springer Berlin Heidelberg, 2012, doi: 10.1007/978-3-642-30561-0_19, pg. 269-287

73. J.Bezivin, Model Driven Engineering, “An Emerging Technical Space, International Summer School, Summer School on. Generative and Transformational Techniques. in Software Engineering”, 4-8 Jul. 2005, Braga, Portugal, Publisher: Springer Berlin Heidelberg, pg. 36-64, doi: 10.1007/11877028_2.

74. E. W. Duggana and C. S. Thachenkaryb, “Integrating nominal group technique and joint application development for improved systems requirements determination”, Information & Management, vol. 41, no. 4, pg. 399–411, 2004, DOI: 10.1016/S0378- 7206(03)00080-6

75. G. Madey V. Freeh R. Tynan, “The open source software development phenomenon an analysis based on social network theory”, Proceedings of the 8th Americas Conference on Information Systems, AMCIS, 9-11 Aug. 2002, Dallas, USA, 2002, pg. 1806-1813.

76. G. Lory, D. Campbell, A. Robin, G. Simmons and P. Rytkonen, “Microsoft Solutions Framework version 3.0 Overview”, White Paper, June 2003

77. G. Pollice, “Using the Rational Unified Process for Small Projects: Expanding Upon eXtreme Programming”, Rational Software White Paper, 2001.

78. Rajeev Sharma and Jitenra Nath Singh, “Systematic Literature Review and Comparative Studies on Human Centred Software Development”, Galgotias University, Greater Noida, 16th-17thDec.2022,4th IEE International Conference on Advances in Computing, Communication Control and Networking,2022 (ICAC3N-22)

79. Hien Nguyen Ngoc, Ganix Lasa, and Ion Iriarte , “Human – Centred design in industry 4.0: Case study review and opportunities for future research”, Jounal of Intelligent Manufacturing, Volume 33, pp 35-76, 11 June, 2021.

80. Omer Uludag, Pascal Philipp, Abheestha Putta, Maria Paasivaara, Casper Lassenius and Florian Matthes, “Revealing the state of the art of large – scale agile development research: A systematic mapping study”, Journal of Systems and Software, Volume 194, December 2022.

81. Blomkvist, J., & Holmlid, S, “Service Prototyping According to Service Design Practitioners. Conference Proceedings”, ServDes. 2010, p. 1-11. Linkoping, Sweden, 2010.

82. Liedtka, J, “Evaluating the Impact of Design Thinking in Action”. Charlottesville, US: University of Virginia, 2017.

83. Milena Savkovic, Danijela Ciric Lalic, Maja Miloradov, Jelena Curcic & Nenad Simeunovic, “Agile and Digital Transformation in Manufacturing: A Bibliometric Review, Current Research Trends and Future Avenu”, IFIP International Conference on Advances in Production Management Systems -2022.

84. Camacho, M, “An Integrative Model of Design Thinking. 21st DMI”, Academic Design Management Conference. London: Next Wave, 2018.

85. Dam, R., & Siang, T, “What is Design Thinking and Why Is It So Popular? Retrieved from Design-IDEO”’ https://designthinking.ideo.com/resources/what-is-design-thinking-and-whyis-it-so-popular, 2020.

86. Clack, L., & Ellison, R, “Innovation in Service Design Thinking,” Service Design and Service Thinking in Healthcare and Hospital Management, p. 84-93. Switzerland: Springer Nature, 2019.

87. Schumacher T, & Mayer S, "Preparing Managers for Turbulent Contexts: Teaching the Principles of Design Thinking", Journal of Management Education, Vol. 42, issue: 4, p. 496-523, 2018.

88. Neil B. Harrison, "A Study of Extreme Programming in a Large Company", Neil B. Harrison Avaya Labs 1300 W. 120th Ave. Westminster, CO 80234 1-303-538-1541 nbharrison@avaya.com, See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/250790483/2003.

89. Anchit Shrivastava1 , Isha Jaggi2, , Nandita Katoch3 , Deepali Gupta4 and Sheifali Gupta5, "A Systematic Review on Extreme Programming", 12345 Chitkara University Institute of Engineering and technology, Chitkara University, Rajpura, Punjab, 140401, India, Journal of Physics: Conference Series 1969 (2021) 012046, IRMAS 2021 IOP Publishing doi:10.1088/1742-6596/1969/1/012046, 2021.

90. Bosch J and Bosch-Sijtsema PM, "Introducing agile customer-centered development in a legacy software product line", Softw - Pract Exp 2011; 41:871–882.July 2011.

91. Ying X. Cai, Jun Liang Lin and Ruxin Zhang, "When and how to implement design thinking in the innovation process: A Longitudinal Case Study", Technovation Journal, 01 Aug 2023.

92. Shenhar AJ, Dvir. Reinventando o gerenciamento "de projetos: a abordagem diamante ao crescimento e inovação bem sucedidos". 1st ed. São Paulo: M. Books; 2010.

93. Sheppard B, Edson J and Kouyoumjian G. "More than a feeling: Ten design practices to deliver business value"https://www.mckinsey.com/business-functions/mckinsey-design/our-insights/more-than-a-feeling-ten-design-practices-to-deliver- business-value, accessed March 20, 2018.

94. Nicolas Rosch, Victor Tiberius and Sascha Kraus, "Design thinking for innovation: context factors, process, and outcomes", European Journal of Innovation Management, pp 160-176, 26 No.7, June-2023.

95. Owens D and Khazanchi D. "From Strategic Intent to Implementation: How Information Technology initiatives take shape in organizations".51st Hawaii Int. Conf. Syst. Sci., p. 4783–92, 2018.

96. Curran C, Garrett D and Puthiyamadam T. "A decade of digital - Keeping pace with transformation". 2017.

97. Mustonen-Ollila E and Lyytinen K, "Why organizations adopt information system process innovations: a longitudinal study using Diffusion of Innovation theory”, Info Syst J 2003; 13:275–97. doi:10.1046/j.1365-2575.2003.00141.

98. Achi A, Salinesi C and Viscusi G. “Innovation capacity and the role of information systems: a qualitative study". J Manag Anal 2016; 3:333– 60. doi:10.1080/23270012.2016.1239228,

99. Aiman Khan, Iqra Zafar Nazir and Muhammad Abbas, “The Impact of Agile Methodology (DSDM) on Software Project Management”,Department of Computer Engineering, NUST, College of E&ME, Islamabad, Pakistan, International Conference on Engineering, Computing & Information Technology, ICECIT 2017, pp:1-6.

100. Dynamic system development method (DSDM) http://www.freetutes.com/systemanalysis/sa2-dynamic-system-development-method.html

101. Katja Thoring and Roland M. Mueller,“The Agile Landscape of Design Thinking”, Elagar Online, https://www.elgaronline.com/edcollchap/book/9781802203134,10 Mar 2023.

102. Jessica Gaulton MD, MPH, Byron Crowe MD, Jules Sherman MFA, “How Design Thinking and Quality Improvement Can Be Integrated into a “Human-Centered Quality Improvement” to Solve Problems in Perinatology, Science Direct Clinics in Perinatology volume 50, Issue 2,Pages 435-448, June 2023

103. Extreme Programming. What is Extreme Programming? [Online] Retrieved 18th March 2009. Available at: www.extremeprogramming.org

104. C. Schwaber and R. Fichera, “Corporate IT leads the second wave of agile adoption”, Forrester Research, Inc, 2005.

105. Agile Alliance. “Manifesto for Agile Software Development”, [Online] Retrieved 16th March 2009. Available at: http://www.agilemanifesto.org.

106. Alberico Travassos Rosario, “Design Thinking in Product Design: Challenges and Opportunities”, IGI Global, Innovative Digital Practices and Globalization in Higher Education, 17 Feb 2023.

107. J. Erickson, K. Lyytinen and K. Siau, “Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research”. In Journal of Database Management, 16(4), 2005, 88-100.

108. F. Keenan, “Agile process tailoring and problem analysis (APTLY). In Software Engineering”, Proceedings, 26th International Conference, ICSE (2004).

109. Crystal Clear software development,Retrieved 22nd March 2009. Available at: http://en.wikipedia.org/wiki/Crystal_Clear(software_development), March-2009.

110. Ashish Agrawal, Malay Tripathi, Sadhana Singh and L.S.Maurya "AGILE: Boon for today's Software Industry-A Review", Master Of Technology, SRMS CET Bareilly CS/IT Deptt., SRMS CET Bareilly, International Journal of Scientific and Research Publications, Volume 3, Issue 12, December 2013, 1 ISSN 2250-3153.

111. Pritesh, "5 Best Agile Project Management Tools And Techniques", https://squeezegrowth.com/agile-project-management-tools-and-techniques,Jan-2022.

112. Feature Driven Development, Online Retrieved 18th March 2009. Available at: http://en.wikipedia.org/wiki/Feature_Driven_Development

113. M. Cristal, D. Wildt and R. Prikladnicki, "Usage of SCRUM Practices within a Global Company", Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on, 2008, 222-226.

114. M. Singh, "U-SCRUM: An Agile Methodology for Promoting Usability". In Ag. AGILE '08. Conference, Toronto, 2008, 555-560.

115. https://www.neonrain.com/agile-scrum-web-development

116. J. Ferreira, J. Noble, and R. Biddle, "Up-Front Interaction Design in Agile Development", In Agile Processes in Software Engineering and Extreme Programming, Springer, Berlin / Heidelberg , 2007, 9- 16.

117. https://www.techtarget.com/searchcio/definition/Agile-Manifesto

118. https://jelvix.com/blog/pragmatic-agile-framework-and-its-principles

119. Mark F. Tannian, "Embracing Quality with Design Thinking", November- 2019, https://link.springer.com/chapter/10.1007/978-3-030-29509-7_13

120. Paolo Ciancarini Shokhista Eragsheva, Mirko Farina, Damir, Mubarakshin and Giancarlo Succi, Agile methodologies between software development and music production: an empirical study, Journal Name: Frontiers in Computer Science, Published in 14 June 2023.

121. Csikszentmihalyi M, "Flow: The Psychology of Optimal Experience". HarperCollins, New York.

122. Manzo,P.: Fail faster, succeed sooner, https://ssir.org/articles/entry/fail_faster_succeed_sooner (2008). Accessed, June 14 2019.

123. “A4Q Design Thinking Foundation Level”, https://isqi.org/us/en/a4q-design-thinkingfoundation-level (2018). Accessed June 14 2019.

124. “The Design Process : What is the double diamond?” https://www.designcouncil.org.uk/newsopinion/design-process-what-double-diamond. Accessed June 14 2019.

125. Cross, N, “Design Thinking: Understanding How Designers Think and Work”, Berg, Oxford ,2011.

126. Luis Corral[1] and Ilenia Fronz[a2] “Design Thinking and Agile Practices for Software Engineering: An Opportunity for Innovation”, 1Monterrey Institute of Technology and Higher Education Queretaro, Mexico lrcorralv@itesm.mx, 2Free University of Bozen-Bolzano Bolzano, Italy, ilenia.fronza@unibz.it Conference Paper · September 2018 DOI: 10.1145/3241815.3241864, SIGITE’ 2018.

127. The 5 Stages in the Design Thinking Process by Rikke Friis Dam, https://www.interaction-design.org/literature/article/5-stages-in-the-design-thinking-process.

128. Camacho, M,“An Integrative Model of Design Thinking. 21st DMI”, Academic Design Management Conference. London: Next Wave, 2018.

129. https://canvas.unl.edu/courses/73802/pages/5-stages-of-design-thinking?module_item_id=1968000.

130. Abdullahi Sani[1], Adila Firdaus[2], Imran Ghani[3], “A Review on Software Development Security Engineering using Dynamic System Method (DSDM)”, [1,2] Faculty of Computing Universiti Teknologi Malaysia, [3]Faculty of Computing Universiti Teknologi Malaysia Seung Ryul Jeong School of Management Information Systems (MIS), Kookmin University, Korea,2013.

131. Saja Al Qurashi, M. Rizwan Jameel Qureshi , “Scrum of Scrums Solution for Large Size Teams Using Scrum Methodology”, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia, Life Science Journal 2014;11(8) http://www.lifesciencesite.com,2014.

132. Torgeir Dingsoyr,Sridhar Nerur, VenuGopal Balijepally, “A decade of agile methodologies: Towards explaining agile software development”, Journal of Systems and Software Volume 85,Issues 6, June 2012,Pages 1213-1221,2012.

133. Enric Senabre Hidalgo, “Adapting the scrum framework for agile project management in science:case study of a distributed reseach initiative”, doi: 10.1016/j.heliyon.2019.e01447, PMCID: PMC6441834, March-2019.

134. DIS I. 9241-210: 2010. Ergonomics of human system interaction-Part 210: Human-centred design for interactive systems. International Standardization Organization (ISO). Switzerland: 2010

135. Wright, G. P, "Success rates by software development methodology in information technology project management : A quantitative analysis" , UMI Number : 3590342, UMI Dissertation Publishing, ProQuest LLC, Michigan, 2013.

136. Dr Taylor Willmott, "Change by Design : The 5 -Step Human Centered Design Process, Adopt a new way of thinking and doing", griffth university, https://www.griffith.edu.au,2022.

137. Juntao Qiu, "A Brief History of Test Driven Development", ResearchGate, DOI:10.1007/978-1-4842-9648-6_1,In book Test-Driven Development with React and TypeScript(pp 1-13), Aug:2023.

138. Tayba Farooqui,Tauseef Rana, Fakeeha Jafari, "Impact of Human-Centred Design Process(HCDP) on Software Development Process", Published in: 2019 2nd International Conference on Communication, Computing and Digital systems (C-CODE), 06-07 March 2019, DOI: 10.1109/C-CODE.2019.8680978, INSPEC Accession Number: 18566945.

139. Cherry Picking, Franziska Dobrigkeit, Christoph Matthies, Philipp Pajak, and Ralf Teusner Hasso,"Agile Software Development Teams Applying Design Thinking Tools" Plattner Institute, University of Potsdam, Potsdam, Germany franziska.dobrigkeit@hpi.de,Oct-2021.

140. Tapish Panwar1*, Kalim Khan 1 Assistant Professor, Rizvi Institute of Management Studies and Research, Mumbai, India. 2 Professor, Rizvi Institute of Management Studies and Research, Mumbai, India, "Integrating Design Thinking in Service Design Process: A Conceptual Review", received: 2020/11/25, Accepted: 2021/05/27, 2021.

141. Rahmin Bender Salazar, "Design thinking as an effective method for problem-setting and needfinding for entrepreueurial teams addressing wicked problems", Journal of Innovation and Entrepreneurship,Published 13 April 2023,Article number: 24,2023.

142. Gruber, M., De Leon, N., George, G., & Thompson, P, "Managing by design", Academy of Management Journal, 58(1), P-7, 2015.

143. Madhup K. Gandhi, Chetan Chaudhari and Vishaka Singh: Study of challenges in agile software development practices in non-government organizations in India. AIP Conf.Proc.2954, 020009, 2023.

144. Sharique Ahmad and Saeeda Wasim, Agile Methodology in Healthcare and Medical Practices: A Narrative Review, Scholars International Journal of Traditional and Complementary Medicine, September 2023.

145. Rick Dove, Kerry Lunney, Dr. Michael Orosz, and Dr. Mike Yokell, "Agile Systems Engineering – Eight Core Aspects", INCOSE International Symposium, 31'st August 2023.

146. Holger Fischer and Björn Senft, "Human-Centered Software Engineering as a Chance to Ensure Software Quality Within the Digitization of Human Workflows Human-Centered and Error-Resilient Systems Development",Volume 9856, ISBN: 978-3-319-44901-2, 2016.

147. Xavier Ferre, Nelson Medinilla, Universidad Politécnica de Madrid, "How a Human-Centered Approach Impacts Software Development", July 2007, DOI:10.1007/978-3-540-73105-4_8 Conference: Human - Computer Interaction. Interaction Design and Usability, 12th International Conference, HCI ,international 2007, Beijing, China, Proceedings, Part I. July 22-27, 2007.

148. Judy Matthews, Cara Wrigley, "Design and Desing Thinking in Business and Management Higher Education", Journal of Learning Design,Vol 10 No-1,2017.

149. Karan Shah, "Agile Adoption and Development Trends for 2023", Solute LABS/BLOG/18 April 2023, www.solutelabs.com/blog/top-agile-trends.

150. Arnold B. Bakker, Daantje Derks and Dylan Molenaar, "Agile work practices: measurement and mechanism", European Journal of Work and Organizational Psychology, Volume 32, Issue 1, Published online July 2022.

151. Julia Yumi Ito, Franciane Freitas Silveira, Igor Polezi Munhoz, and Alessandra Cristina Santos Akkari, International publication trends in Lean Agile Management research: a bibliometric analysis, Procedia Computer Science, Volume 219, Pages 666-673, 2023.

152. Irene Gottgens 1, Sabine Oertelt-Prigione, Strode, "The Application of Human-Centered Design Approaches in Health Research and Innovation: A Narrative Review of Current Practices", Department of Primary and Community Care,Radbound University Medical Center, Nijmegen, Netherlands, JMIR Mhealt Uhealth, 9(12) : e28102. Doi:10.2196/28102, 6th Dec-2021.

153. Naveen Kumar Reddy and SRIT Proddatur, "Design Thinking-Evolution & Its Importance In Business", Indian Scientific Journal of Research In Engineering And Management,20 March 2023.

154. Edgar Brea, Ida Someh, Emma Freya, Brett Thebault and Shazia Sadiq, "Integrating Design Thinking and Agile Approaches in Analytics Development The Case of Aginic", Sage Journals: Jounal of Information Technology Teaching Cases, May 25, 2023.

155. Pusca, D. & Northwood and D. O, "Design thinking and its application to problem-solving", Global Journal of Engineering Education, V. 20, No. 1, 2018.

156. Pavie, X., Cathy, D, "Leveraging uncertainty: a practical approach to the integration of responsible innovation through design thinking", Procedia-Social & Behavioral Sciences 213.1040-1049, https://doi.org/10.1016/j.sbspro.2015.11.523

157. Docherty.C, "Perspectives on Design Thinking for Social Innovation", Design Journal, 20 (6), p. 719-724, 2017.

158. Teixeira, J. G., Patrı´cio, L., Huang, K.-H., Fisk, R. P., No´brega, L., & Constantine, L. "The MINDS Method: Integrating Management and Interaction Design Perspectives for Service Design", Journal of Service Research, 20(3), p. 240-258, 2017.

159. Stone, B, "The Upstarts: How Uber, Airbnb, and the Killer Companies of the New Silicon Valley Are Changing the World. Boston", Brown and Company, 2017.

160. Paolo Ciancarini Shokhista Eragsheva, Mirko Farina, Damir, Mubarakshin and Giancarlo Succi, "Agile methodologies between software development and music production: an empirical study", Journal Name: Frontiers in Computer Science, Published in 14 June 2023.

161. Human-centered Software Development build stronger software by designing for human interaction, Author: Edward Park, Senior Software Engineer, https://medium.com/headspace-engineering/human-centered-software-development-3b0aa77f8897, May 28, 2021.

162. Luis F. Mendivelso1,2, Kelly Garcés1* and Rubby Casallas1, Mendivelso et al, "Metric-centered and technologyindependent architectural views for software comprehension", Journal of Software Engineering Research and Development 6:16 https://doi.org/10.1186/s40411-018-0060-6, 2018.

163. Bagnato A, Sadovykh A, Dahab S, Maag S, Cavalli A, Stefanescu A, Rocheteau J, Mallouli S, and Mallouli W, "Modeling omg smm metrics using the modelio modeling tool in the measure project", 2017.

164. Stevanetic S, Haitzer T, and Zdun U, "Supporting software evolution by integrating DSL-based architectural abstraction and understandability related metrics", ACM International Conference Proceeding Series. ACM, Austria. https://doi.org/10.1145/ 2642803.2642822, 2014

165. Rahmin Bender-Salazar, "Design thinking as an effective method for problem-setting and need finding for entrepreneurial teams addressing wicked problems", Journal of Innovation and Entrepreneurship 12, Article No. 24 (2023) Published in April 2023.

166. Rajeev Sharma, J.N Singh, "An Intelligent Human Centred Software Development Framework" Journal Name (International Journal of Information Science and Applied Engineering (IJISAE). Scopus), ISSN: 21476799, Vol 11 NO. 10S (2023).

167. Rajeev Sharma, J.N.Singh, "Human Centre Software Development Model and User Acceptance Index Factors Count", International Conference in Multidisciplinary Concepts in Management, ICMCM-2023, Glbjaj Greater Noida, June 30 & 1 July 2023.

AUTHOR’S PROFILE:

Mr. Rajeev Sharma, is currently working as Assistant Professor, in School of Computing Science & Engineering, Galgotias University. He has total experience of more than 15+ years of teaching and around 1+ year experience of Industry and has been associated with Galgotias Education Institutes (GEI) since 2008 and associated with Galgotias University since 2017. He completed MTech in CSE, MCA, MPhil in CSE, and MA in Economics. He is firm believer of sincerity and efficiency in work and he always adopt a noble work ethic with the ability to excel in fast-paced, time-sensitive environment and always focus student-oriented teaching methodologies. Being a passionate teacher, he believes that teaching is not merely bound to making the students understand the underlying concepts of a subject but also to develop critical thinking, experience and evaluate alternate approaches for problem solving. He always put his efforts towards the overall developments of his students. He has convened/organized more than 10 co-curricular/Extracurricular events for the benefit of students in the past 5 years. He organized one zonal state level sport fest under AKTU Lucknow. His research is Human Centred Software Development (HCSD). His areas of interest software engineering, agile engineering, design thinking (DT), human centred software development, computer networks. He completed certificates in areas of higher teaching methodologies from NPTEL, TALE and OBE. He attended many workshop and faculty development programs on Data Analytics, AI&ML, OBE, Start-up and Innovation areas conducted by NPTEL, NITTTR Bhopal and Chennai.He is the active members of AICTE institute innovative council, Program Chair in CSE & alumni coordinator of GPTC, member of SDG committee of Galgotias University. He is doing heartfulness practices since 1996.