Journal of Computer Networks, 2017, Vol. 4, No. 1, 30-47
Available online at http://pubs.sciepub.com/jcn/4/1/4
©Science and Education Publishing
DOI:10.12691/jcn-4-1-4

Towards an Ameliorated Approach for Design and Maturity of Cloud Service Technical Activities and Cloud Project Management by Overcoming the Service Scope Creep

Manu A R[1,*], Shivanand M Handigund[2], Manoj Kumar M[3], Dinesha H A[1], V K Agrawal[4], K N Balasubramanya Murthy[1], Nandakumar A N[5]

[1]Department of Information Science and Engineering, PES Institute of Techonology, VTU, Bangalore, India
[2]Department of Computer Science and Engineering, M Tech, Vemana Institute of Technology, Bangalore, India
[3]Jain University, Bangalore, India
[4]Department of Information Science and Engineering, PES University, Bangalore, Karnataka, India
[5]Department of Information Science and Engineering, New Horizon college of Engineering, Bangalore, India
*Corresponding author: manu.a.ravi@gmail.com

Abstract In general the end service/product devised is offered as a service over cloud. The service is coded as a software service. Whereas the service that is coded as software project is time bounded endeavor relating the scope, risk, time schedule, cost investment, effort, in addition it involves human and computing resources. The service is devised as technical activities (TA's) based on managerial activities as put forth by PMI. These managerial activities enhance the value of quality parameters with varying art. . The au-courant velocity of the crowd computing service software design and development is derisorily low due to invertible use of cloud service managerial activities devised by PMI. The clairvoyant study reveals that the TA's do not have proper predefined methodological existence to execute the service TA's, it is under dogmatic effect. In the absence of multilateral experts for the service project management activities (PMA's) devised by PMI, US is de-facto standard is used to upgrade the service quality core activities. Unfortunately the PMA's to devise the core basic TA's is under derision of foppishness. The multilateral collaborative stakeholders are ignorant, without proper vision, objective mission, it has become an ornamental show that there is a urgency and need to devise the core TA's for developing of cloud service software. The TA's are abstracted from by-products of the managerial activities. But this activities of cloud service development are devised by neglecting the core TA's. There is no pari-passu between these two activities of devising the secured cloud services. Devising the TA's and MA's involves filling the tables devised by MA's with the project service inputs without derision of foxiness. Since both TA's and MA's take same input and output base. The PMA is devised using 2-D space devised by 5 PLC's, and 11 KA's. The MA's involves enhancing the TA's quality parameters. Connecting to the computing system any service using any network, any place, anywhere, by any one at any time using any device by connecting to internetwork. Normally the secured end product of cloud computing system project service that develops the computing system software is a time bound endeavor involving scope, risk-schedule, human and system computing resources. Normally in cloud system SDLC, engineers and stakeholders adopt the secured technical activities based on the computing system managerial operational activities. These activities are designed by Software Engineering Institute (SEI), NIST and Project management institute (PMI). These are based on tables, metrics, standards, guidelines, rules is referred in secured computing systems managerial activities are used to identify the activities required for the ensuing secured software this is termed as computing system service technical activities. The secured computing managerial activities which enhance the ideals of these QOS factors are supported on shifting art by means of the computing system solid waste division of the crowd computing system; we explore the project scope management [6,7]. Using the computing system solid waste division of the crowd computing system, we explore the project scope management. Further we explicate scope creep versus progressive amplification, and discuss the common causes for scope creeping of the computing system product/project service scope creep management. Some steps to prevent the scope creep, using the case study and present the lessons learned from the case studies.

Keywords: *cloud service project scope creep, cloud service project management, cloud service management, cloud service web crawlers, and cloud agility*

Cite This Article: Manu A R, Shivanand M Handigund, Manoj Kumar M, Dinesha H A, V K Agrawal, K N Balasubramanya Murthy, and Nandakumar A N, "Towards an Ameliorated Approach for Design and

Maturity of Cloud Service Technical Activities and Cloud Project Management by Overcoming the Service Scope Creep." *Journal of Computer Networks*, vol. 4, no. 1 (2017): 30-47. doi: 10.12691/jcn-4-1-4.

1. Introduction

The au-courant success rate of cloud computing system software development rate is derisorily short. The root grounds for this short success rate is the invertible use of managerial activities developed by PMI for traditional computing systems it don't suits to present day Cloud computing systems. The study reveals that the managerial and operational activity that does not exists on its own. Also, there is no standard methodology to carry out technical activities which do not exist on its own. There is no clairvoyant methodology to carry out technical activities of the cloud computing system service project. The use of managerial activities to derive technical, operational, and managerial activities in cloud computing ecosystem is small sauté the effects of the technical activities. In absence of the expert choose cloud service project management activities developed by PMI USA as de-facto standards [1] are used to enhance the quality of the core activities. Unfortunately at present many of the companies are using their own cloud system managerial activities to develop the core activity this is derision of foppishness. People are ignorant of utilizing their own vision, mission, and objective [7,8]. It has become ornamental that means they should use to develop the technical core activities since these are business and domain project dependent activities. The technical activities are abstracted from the by-products of managerial activities, thus the cloud service development projects neglected the correct core methodology for the development of the technical activities. Even there is no pari-passu between these two types of activities. Developing the TA's based on the managerial activities, means filling up the tables developed by the managerial activities. With the service project input, these may at most develop the program not the software this is derision of foppishness. Both technical activities and managerial activities take same input and output base. These PM activities have been developed in 2-Dimensional space formed by 5 Cloud service project life cycle (PLC's) and eleven knowledge areas (KA's). The managerial activities enhance the quality parameter values. The enhancement needs existing quality parameter value which is attained by technical activities [9,10,11]. Currently there are no automated tools, techniques and methodology for the development of the technical activities required for CCS software development. This unit contains our developed methodology for the core activity based on the objectives and input feature. Managerial activities and technical activities are dimensionally orthogonal to each other; both should be transformed to the single dimensional. The only methodology is to transfer it to the 3- dimensional, we are blending it together. As per the PMI glossary published PMBOK guide in 2008, and recent publications, Project scope management is defined as it inclusive of the process tasks related to make sure that computing systems project/product service. It includes the total work effort required by, and only the work effort required to complete the project/service successfully.

Scope creep (dimensional crawling): accumulation of features and product service functionality. Scope is devoid of addressing the result on rate, time schedule and computing resources and entity effort or without benefactor endorsement. Progressive elaboration with continuous integration and continuous development using dev-ops, ops-sec, sec-ops, agile methodology for continual improving using iterative comprehensive plan and more precise estimation prepared and availed as the computing system evolves. The system progression in the long run with its output of the exact and computation plans with an outcome of the product service with successive iterations of the forecasting progression. Scope creep is managed through the AD cover replacement and refactoring, with extended time duration to execute design services and fabrication management to improve the AD [12].

The main reasons for Scope creeping of the product/project service are due to derisory project planning related to: communication management, DR, risk and quality management. Unproductive PM: Stakeholder management, weakly documented scope, poorly managed service scope, undocumented exploratory [13,14] and ad-hoc postulations, fruitless monitoring and control procedure, processes leads to scope creep of the project management.

1.1. Preface

The cloud computing system project service management is buzz word in IT industry dealing with crowd computing systems like cloud service projects. Nowadays the success rate is pathetically low. The clairvoyant study [15,16] for the root cause analysis indicates that the software industries blindly consider the managerial activities developed by PMI as podium for the core activities i.e. technical and operational activities [17,18]. At present the academic and industry researchers [19,20] have not recognized precise technical activities obligatory for secure architectonic design and cloud service project management as their concept is based on individual project/service dependence i.e. on vision, mission, and the objectives and the input features.

1. To apply the managerial control on the architectural design and TA's should be in place, currently without the technical activities they are using the [21,22] quality enhancement managerial activities to produce basic quality requirements. This paralogizes the CCS software development process,
2. 2ndly the stripped bare PMA are also not automatable, this adds to floundering struggle through the software process. Secondly most of PMBOK secretarial actions are not right for computing system software (s/w) development projects as the end software product is intangible. This necessitates the presence of SDLC stages.
3. 3rdly the realization of most of the tools and techniques suggested for PMBOK managerial

activities are left out as an expert judgement these methodologies have no scope in engineering [23,24,25].

4. 4thly the number of technical and managerial activities is asymptotically bound by order of O(nm) therefore for blending these two types of activities there exist a necessity of decomposing the activities to micro and nano sized actions enable them to be free from the dimension [26,27],
5. 5thly the project service involves gestalt of technical activities developed in SDLC stages (tightly coupled activities) since these activities determine the state-of-art business process, and attain the quality parameters [28,29,30].
6. To avoid the scope creep due to management activities [31,32,33].

We are reverting the para-logized process with architectonic process in viz. In our case we are trying to build the methodology for the (abstraction) conceptual notion of technical activities abstracted from diverse input characteristics and QOS objectivities. Then blend the technical activities with the appropriate managerial activities as they are made dimensionally same [34,35].

Vision of this work is to eliminate embezzlement of managerial activities to generate technical activities. Mission includes designing and developing methodology for the abstraction of technical activities from the objectivities and inputting feature and blending them with equipollent managerial activities.

Objectivities include:

- To develop technical activities through the establishment of correspondence between the semiotics of input feature and objectives [36,37,38].
- To form blend of equipollent managerial and technical activities[39].
- To design and develop the reticulation of activities for CCS software projects.

The main motivation of this work is based on PMI though have developed well defined managerial activities in the point value from 2-D pace viz. PLC and KA. It does not help to identify technical activities required for the software projects. If the technical activities are derived from the table generated by managerial activities, there exist many pitfalls which can be witnessed in the au-courant CCS software development scenario. The recession in the software industries is a bottleneck in the utilization of software in all stages of CCS computing life cycle.

The very purpose of CCS professional managerial administrative activities is to augment and boost the excellence valued quality features of technical activities in place. Most of the managerial activities are left for multilateral expert judgment, it is an art. Technical activity is an engineering art should be transferred into engineering activity via science but these are not possible as art is not stable, there is no unique art to develop business process. The technical activities cannot be engineered with these types of volatile art; moreover the third pitfall is that both are dimensionally orthogonal to each other, which ruins either the bones of managerial activities or bones of technical activities. The fourth pitfall is that developing the technical, activities based on managerial activities means filling up the tables developed by the managerial activities with the project input these may utmost develop the program not the computing system software. The fifth pitfall is that PMBOK activities are mere managerial activities whose adoption is to enhance the quality of technical activities. The technical activities for other (other than software development) projects are fewer micro service oriented architecture design and loosely coupled CCS services. In such cases, the use of PMBOK [1] managerial activities in their entireties may be irrecusably essential task. For software development projects, the technical activities are either nonexistent and are currently carried out by arbitrary human skill with adherence to managerial activities. The technical activities are voluminous, project specific and are according to SDLC stages. So for blending both types of voluminous, each activity should be made dimensionally equipollent. This is possible only if the correspondence between PLC phases and SDLC stages maintains atomicity of activities or the decomposition is on common principles [30,40].

Managerial Activities

$$\bigcup_{i,j=1}^{m}(PLC\ Phase_i \cap Knowledge\ Area_j) \qquad (15.1)$$

Technical Activities

$$\bigcup_{i,j=1}^{m}\left(SDLC\ Stage_i \cap Knowledge\ Area_j\right) \qquad (15.2)$$

Operational Activities

$$\bigcup_{i,j=1}^{m}\left(\begin{array}{l} SDLC\ Stage_i \cap Knowledge\ Area_j \\ \cap Sytem\ Lifecycle_k \end{array}\right) \qquad (15.3)$$

Technical/Managerial Activities

$$SDLC\ Stage_i \cap PLC\ Phases_j \cap Knowledge\ Area_j \quad (15.4)$$

i.e.,

$$\bigcup SDLC\ stages,\ j \in \bigcup PLC\ Phases \qquad (15.5)$$

Next most of PMBOK cloud system administrative actions are not appropriate for cloud computing system software development/implementation of service product/ projects as the end software product is intangible.

Sixthly the number of technical and managerial activities is asymptotically bounded by O (n) therefore for blending these two types of activities there subsist a necessity of decomposing the activities to enable them to be free from dimension. The project involves gestalt of technical activities developed in SDLC stages (tightly coupled activities). Since these activities determine the state-of-art business process.

The configuration management place vital role in its maintenance. Therefore, technical, management and operational activities forces all the activities need to be dimensionally same. This requires the composition of activities in different level of hierarchy. Determinations of activities along with their unit are to be blended together. Each management activities of PMI cannot be applied as they are holus-bolus and are partially carried out with technical activities which are scattered in different stages of SDLC. Lastly there is a need to determine managerial and operational activities for the CCS service software development projects which are equipollent to technical and operational activities. Thus the technical activities are determined purely based on objectivities and input features. The semiotic gap between these two paves the way for design of technical activities. The equipollence between the activities can be created through their decomposition and reorganization with measureable unit activity. This is written as

CCS Operational/Technical/managerial Activities

$$CCS\ SDLC\ Stage_i \cap CCS\ PLC\ Phases_i. \qquad (15.6)$$

The managerial activities cannot be effectively blended to the technical activities as all the attributes are dimensionally orthogonal to each other [42,43]. The dimensional equipollency is to be established before blending the two types of activities, these above pitfalls/lacunae's motivated us to develop first methodology for the abstraction of technical activities based on the input features and the project objectives and then appropriate equipollent managerial activities are to be blended with technical activities. On the other hand, with clairvoyant study of PMBOK managerial activities, only the suitable activities for software development projects have to be identified and hiatuses created by unsuitable activities [44]. The managerial activities have been developed in to 2 dimensional spaces of PLC phases and knowledge areas (KA's). To optimally enhance quality of the project, the dimensional equipollency between both types of activities is established and each pair of pari-passu activities are blended together to form project activities. This means the technical activities are to be decomposed to be amenable for PLC's and KA's and the managerial activities are to be decomposed to suit SDLC stages.

2. Related Work

In literature review we can find that [1] PMBOK guide published by PMI US, has been well thought-out as a naïve de-facto standard norms and actions with 42 de-facto benchmark paradigm actions seeded beside with their plan II/O processes, with automation tools, technologies and techniques. These activities are amplified evenly all over the 2-D space fashioned by KA's and PLC segments as endorsed in PMBOK. The treatise is fit for common traditional computing project activities but do not suits for secured cloud computing system service projects. Furthermore the practices used to comprehend, the majority of the actions are based on connoisseur multilateral expert judgements whose victory depend on expert's in-depth insightfulness (profundity) and cannot shape the tactics (methodology). This demands a systematic amendment in PA's suiting the cloud computing systems service project management. Moreover, PMBOK has identified only the general activities and not project TA's. In this work an attempt is made to design the software project technical activities and associated managerial activities and placed them in 3-D space formed by 10 KA's, 5 PLC's and 7 SDLC's. An attempt is also made to abstract software PA's through the cloud service project inputs and resulting goal oriented objectives of the computing system project.

In reference [2] authors ascribe The general software service project management as ascribed in real-world existing guide by PMI in general has acknowledged only some of the PA's. But it is stressed out merely on the professional managerial and administrative aspects of the software PA's i.e. on software development project aspects of traditional software development projects but not on TA's aspects. But these do not suit for present day complex cloud computing system mission activities and not on the methodological aspects. The actions offered are also too general to be developed and executed. In addition, the authors of PMI guide have not recognized the automation tools and techniques for automating the management processes for the identified activities in [2]. An author of [3] has described their experience of software system PM in the form of book chapters. Authors [3] well thought-out PA's on computing system PLC and SDLC hypothesis suiting the information system. The info computing system maturity mission forms a division of computing system venture service. This does not suit the service project undertaken from the market demands, where the legal necessity is considered for development of firmware type. Henceforth the [3] cannot be applied for current modern computing system projects. Cloud computing service mission/system supervision is an incorporated scaffold has offered their episode/event occurrence on traditional computing system PM in the form of guiding channel. But this manuscript well thought-out of PA's on the SDLC hypothesis which might perchance outfit to computing statistics scheme of designing and development of computing system projects. This traditional information systems maturity of the project outlined is very little subset of computing system software assignment. This may perhaps not be handy to the service mission contracted and executed from CCS market demands, with legal obligation. In which the propose and expansion of a sort of virtual ware, firmware, software, hardware as a service is considered, consequently this book conceptions cannot be practical and functional for cloud computing software projects.

In book titled "Managing information technology projects", has recognized the traditional PMA's and positioned them into 2-D space shaped by system development life cycle, security life cycle, SDLC and PLC's which is poor for on top of contests. On the other hand it pitches radiance on the panorama of the actions.

In work titled authors "SPM in practice" discussed active IT engineering practices on usual SPM in practice in his employed company. Since precise engineering industrial skill cannot be universalized, due to each engineering business have its own executive organizational practices, processes, resources assets, benefits, sources and CMM level. Additionally, most of the actions of IT engineering trade are data rights secluded and protected. As a result the performance actions of individual corporations cannot be globalized to the cloud computing business computing software venture. Hence the script is not helpful for abstracting business autonomous PA's (project activities).

In the manuscript titled "software Project Management", authors [52], here the authors has identified and acknowledged some of the performance actions and activities according to the book. But the author has not formed the topology of activities/actions and has not positioned in any of the project related space. More over no technical details are available for these activities.

The Prototype: It is utilized the approaches of PMBOK (Project management body of Knowledge) published by PMI it contains 47 de-facto standards and allied activities. IT presents MA's in 2D space of 11 KA's, 5 PLC's phases. These are in very general in nature and its subsets suits for software development activities. An attempt is made to devise TA's and MA's of the computing software activities (SA's), and place them in 5 PLC's and 7 SDLC's. Am attempt is made to spot the service relevant and inappropriate MA's to suiting the software developing cloud computing systems from PMBOK PA's. The authors of [2] as spotted few PA's but stressed only on MA's for SDP's but not on TA's. The actions as discussed in [2] it is to general to implement it. The author has failed to locate the automation tools, techniques, for the spotted activities. In his book the author stresses on TA's for SDP's for identification of TA's. The author of [5] has located PMA's using the 2D space of SDLC, and PLC is not clearly defined and distinguished. Pragmatically the PLC's enclosed in SDLC. PLC and SDLC are orthogonal to each other. We may try to blend the TA's with MA's and place them in 3-D space using 10 KA's, 5 PLC's, and 7 SDLC's. Authors of the references from [6] to [38] provides good source of the work on various aspects of soft engineering aspects project management and blending activities to ravel software engineering aspects beginning from SRS to testing the software engineering management. It presents the engineering management, SRS design, OOAD concepts, UML diagrams etc. References [39] to [50] helps to refer as guidelines to blend the various software engineering activities, it serve as source to references to devise good model for project management.

2.1. Taxonomy

Procedure to control the scope creep:

The core secured Cloud computing systems (CCS service) implementations are designed and managed using service SRS and SLA as the fundamental supporting document for service design. SRS and SLA's generally comprises of the particulars of the secured implementation for the computing system. Through which appropriate secured service implementation can be designed, and developed and/or implemented at root level like architecture and design of the computing system. Securing at design and at architectural level makes the computing system more secured at root level, and helps to plan proper security and service product/project management at each layer of the complex cloud computing system. This details are potentially viewed with various pragmatics like computing tasks/jobs/work, job processes or use-cases respectively. The use-case is exclusive viewpoint whose summation total forms the tight security of the computing system[45]. This work, presents an effort made to abstract useful components from statements of all actors from unformatted unstructured SRS text. Though object oriented paradigm offers a boon to facilitate the development of the information system. It is a common observation that there is no single concrete methodology to design the scope creep management of the CCS project system. As a result, the development of the information system using scope creep management paradigm is a more of a human skill dependent than of systematic methodology. This research aims to bridge gap between the available SRS and the design of system through abstraction of object methods and useful components of activity diagram, sequence dia, class dia, and work process dia [46,47].

One should define and narrate in the documentation of the crowd computing project/service objectives by collating and collecting the service requirements documentation, SLA requirement management planning. In addition, Scope management plan aligned with project scope: Project/product objectives, service SLA requirements, QOS standards and procedures, crowd computing organization premeditated objectives. Also by performing the periodic service review of the service boundary base-lining using logs and service monitoring and controlling, based on project/product performance monitoring using variance analysis, and earned rate administration techniques. Apart from the above techniques it can also controlled using CI-CD integrated change control process, by reviewing, approving and or rejecting, by documenting and implementing the changes to scope dimensionality PM planning. By verifying the scope for formal receipt of completed CI-CD deliverables if rejected reason is documented for non acceptance. If in case any change requests are made also documented with proper notes and reason for the change requests made using punch lists. Upon the service/project closure phase final tasks preformed are project scope review, closure documentation, and final agile stand-up meeting with scrum (retrospection) meeting made to post-mortem lessons learned through out project service execution.

In Crowd computing system involves multilateral change control board which is formally represented all the multilateral stakeholders. They are accountable for all the activities involving estimating, verifying, validating, approving, waiting, or halting, or declining, changes to the project service, with all the CCB recorded multilateral decisions and proposals. Scope substantiation (SS) and QOS control: SS is primarily agitated with receipt of deliverables. Whereas, QOS control is major concern with correctness executed before SS is started.

Figure 1 shows the growth of Scope creep with respect to the tasks, budget and total duration in years plotted with range of 5 years duration.

*SRS: software requirement spec*ification is a record document authored in English language consisting detailed information in generally prepared by the client patron corporation including detailed overview of the computing system. It consists both functional and non-functional service requirements of the systems, the interacting actors interfaces, and prototypes, rules, POC's and service constraints information etc [48]

***The Syntactic taxonomy*:** The syntactic rules communicate the symbolic connotation in UML. The syntactic taxonomy encompasses model elements and its symbolic representations [49].

Semantics: systematizes the symbolic models and its elements into an evocative cluster according the semantics rules. It systematizes diverse signs into a significant entity as a ingredient of pragmatics [50].

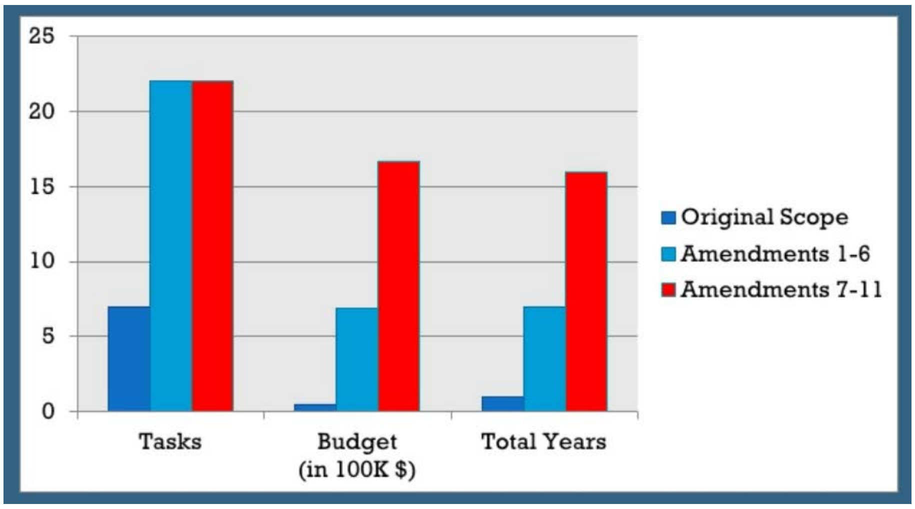

Figure 1. Scope creep with respect to the tasks

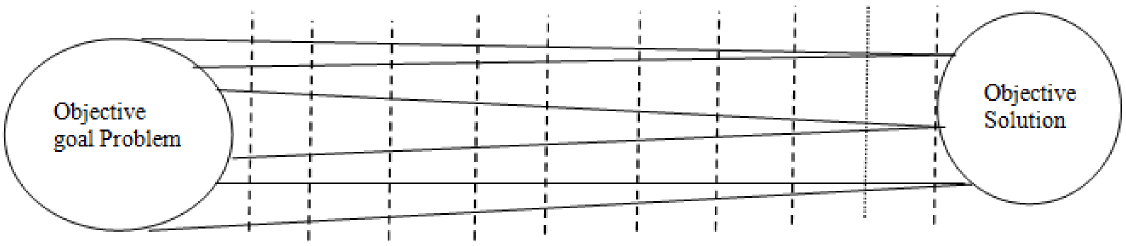

Figure 2. Objectives

Semiotics: Each idiom lingos (visual and pictorial verbal or programming) has able-bodied definite syntactic, semantics, and pragmatics.

The Pragmatics: practical are phrased to be current practices.

Project Service Vision: it is service foresight, of the premeditated service management prudent for future without any actions and activities involvement.

Service Mission: It is the specific job/task/process or purpose needed to realize the quoted vision [51].

Objective goal: It is broad service framework of realizing the activities of the mission as noted in Figure 2.

Service Strength: The factors of the characteristics which facilitates to ship input syntactic in the forward direction to objective syntactic.

SDLC (Software/System) Development Lifecycle: An architectonics design of actions (to be acceptable by the mission) inhibited in ordered junctures of the CCS software plan and maturity. It connotation is restricted and progression of computing info system.

Project service Life cycle phases (PSLC): These are bouquet of behaviour used in assorted progressive stepladders of the computing system projects. This SPLC stages viz. instigating, schedule and forecast planning, testing, concluding, supervising, and domineering.

Skill and Knowledge areas (KA's): Based on the task the PM is grouped into diverse KA's for cloud computing software development project services, the actions are bunched in 10 KA's Viz. computing system Project service integration management, cloud service project scope and dimension management, cloud service project time schedule management. In addition, computing system project cost and outlay pricing and billing management, computing system project quality of service management, computing system project Human resource and computing systems resource management, project service communication systems and network management, project service risk, threat and security management, and project/service resources procurement management. Also, project service or system configuration management etc.

PA's (project activities) conceded out throughout the evolution of the project service to accomplish the associated vision, mission, and its assorted objectives, this may comprises of technological, operational, and managerial activities.

Activity: The business progression consists of succession of assorted actions/activities. It is principal entity that is budding to partake in use cases, work /jobs/tasks/ procedure and effort [26].

$$\begin{aligned}&\text{Activity}\\&= \text{Work / jobs / tasks progression} \cap \text{Use cases}_i \qquad (15.7)\\&\quad \cap \text{ Work process } (\text{where work in SDLC stages}).\end{aligned}$$

2.2. Projected Methodology

2.2.1. Problem Proclamation of the Computing System Project

To plan, design, and develop to build an ameliorated methodology for the discovery of cloud service PA's comprising and involving the blend of technological and professional administrative activities.

Input: Software requirement Specs (SRS) together with visualization, mission, and objectives of the computing project service.

Output: total reticulation of TA's as specified absolutely in cloud service objectives.

Tools Devised:

- Directed control flow table (CFT)
- Data info flood table (DFT)
- Synonymies
- SWOT (Strength weakness, Opportunities, threats)

2.2.2. SRS

An SRS is an absolute explanation of the anticipated rationale function and ecosystem/environment for computing system software under consideration for development. The SRS also in general contains vision, task and objectives of the computing system project [33].

Summary of the proposed methodology: In this project we are aiming to develop a methodology for the design of technical activities from the input features and objective. The client provides the SRS to the strategic manager, abstract the vision, mission and objectives. From objective abstract the semiotics. Then identify the synonymies from the input features, then form the closure from the input to objective feature then perform the SWOT analysis by considering the prototype, then blend the technical activities with the appropriate managerial activities [34].

Functional requirements [35]:

- To abstract semiotics from each objectives
- To identify the synonymies base of the objectives,
- To abstract referenced –defined attributes from the input,
- To design CFT using ref-def table,
- To design DFT in the current flow order table (CFT).
- To identify the synonymies base of the input features,
- To perform closure operation from input feature to objectives,
- To perform SWOT analysis,
- To blend the technical activities with the appropriate managerial activities.

Non functional requirements (NFR) [30]:

At this occasion it is trialled to substantiate the QOS factors in CCS PM and to silhouette the reticulation of the PA's.

Enterprise environmental requirements:

Every project is affected by some external factors time constraints.

Operating environment requirements: The clairvoyant logic reveals that their does not exist any methodology to carry out technical and managerial activities of the project. In the absence of experts choose the project management activities development of PMI USA as de-facto standards for the software development project service process. These PMA's have been developed for the general purpose projects. These activities have been developed in 2-D space formed by the live project life cycle phases (PLC's) and 11 KA's. Normally, the management activities blended with the appropriate technical activities enhances the quality of technical activities. The mere use of the managerial activities with technical activities and without dimensionally blended results in derision. Presently this is the au-corant scenario in all our IT industries. Even these administrative actions are not absolutely apposite for cloud computing system software development service projects. Moreover many of the tools and techniques are left to the expert's judgement which does not indicate the methodology or procedure. No one has developed the technical activities required for the software development project. There is a need to develop these technical activities based on the objectivities and inputs features, and blend them with the appropriate managerial activities to make them dimensionally equipollent to each other.

Performance requirements: To illustrate the software project management suffers from the lack of technical activities and passive managerial activities. This hindrances result in pathetically low success of the computing software projects, the lacunae exists, because the CCS software engineering concepts are not developed on strong foundation of mathematical rigor. CCS Software engineering concepts are not developed on strong foundation of mathematical rigor. Software project managements are reselected with introjections of the technical activities and appropriate refinement of managerial activities [51].

Standard Requirements: The standard requirements for the abstractions of the technical activities are: 1. we have to consider the objectives for the abstraction of the technical activities.

For the activities particular input and output need to be identified.

2.3. Quality Parameters in Software Project Management

Existing system could not satisfy quintuplet (correctness, completeness, efficacy, efficiency and robustness) features software project management.

Correctness:

Existing system:

At present they are developing technical activities (TA) based on the managerial activities.

Managerial activities are compared to changing art there is long way to change from art to engineer, from this volatile art it cannot be transformed to engineering art. So we can't derive the technical activities.

Managerial activities for the general projects are in the tabular from; they are trying to fix the technical activities to this tabular form.

Justification:

We are developing the tactic for the discovery of TA's first for these technical activities we are refining the managerial activities.

Completeness:

- Existing system: the project is complete if it meets all of its objectives. Currently objective has also because ornamental there is no process to authenticate the completeness of the project [52].

Here in our work we are identifying the technical activities based on the objectives and the input features.

Robustness:

Existing system:

In the existing system there is absence of the reticulation of the blended activities.

Justification:

We are trying to reticulate the activities and blend them with equivalent dimensional equipollence.

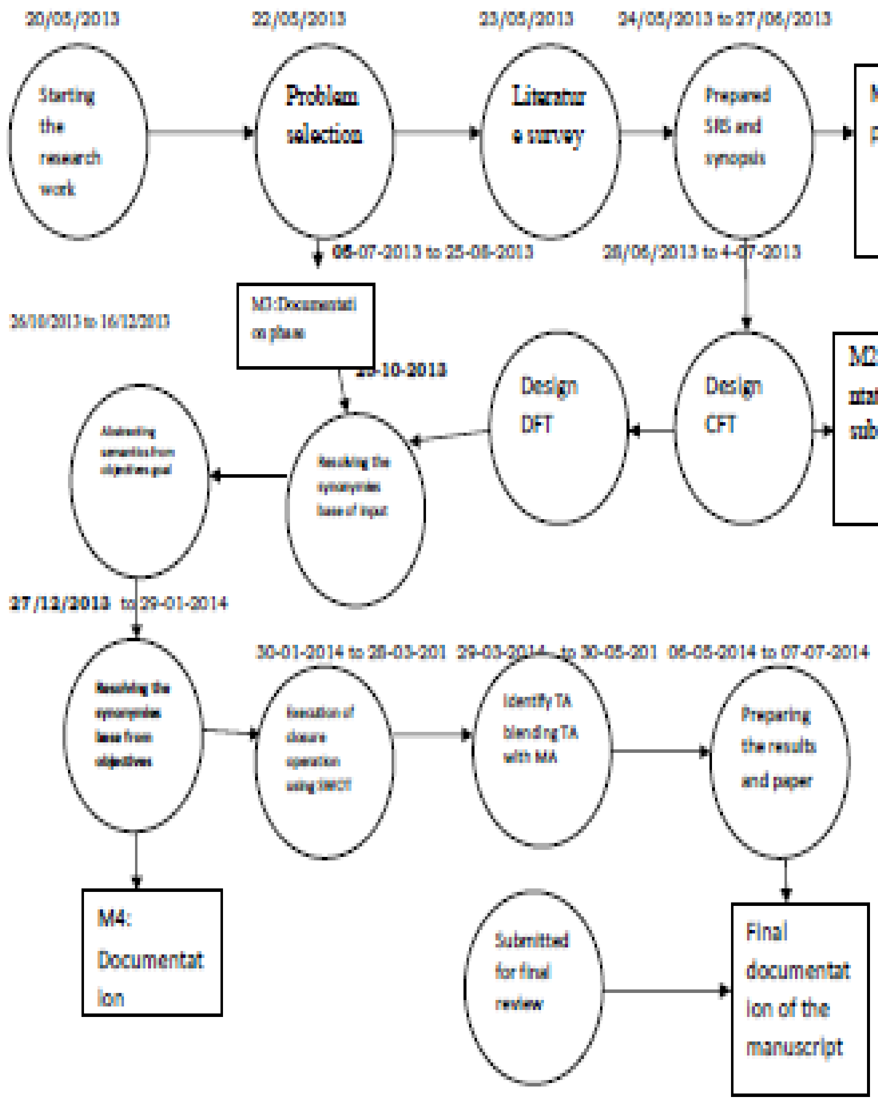
20/05/2013
22/05/2013
23/05/2013
24/05/2013 to 27/06/2013
Starting the research work
Problem selection
Literatur e survey
Prepared SRS and synopsis
M1-1st phase
08-07-2013 to 25-08-2013
28/06/2013 to 4-07-2013
26/10/2013 to 16/12/2013
M3:Documentati on phase
25-10-2013
Design DFT
Design CFT
M2:Documentation and submission
Abstracting semantics from objectives goal
Resolving the synonymies base of input
27/12/2013 to 29-01-2014
30-01-2014 to 28-03-201
29-03-2014 to 30-05-201
06-05-2014 to 07-07-2014
Resolving the synonymies base from objectives
Execution of closure operation using SMCIT
Identify TA blending TA with MA
Preparing the results and paper
M4: Documentation
Submitted for final review
Final documentation of the manuscript

Figure 3. Activity Diagram

Existing system:

Technical activities should be decided based on the objectives, at present they are developing the technical activities based on the managerial activities it is superficial hence the efficiency depends on human skill.

Efficiency:

Existing system:

They are adopting wrong policies, without quality.

We are trying to enhance the quality.

Technical activities and managerial activities are dimensionally orthogonal to each other, which ruins either the bones of managerial activities or bones of technical activities.

3. Procedure to Mould Cloud Computing System SRS

Cloud computing Software requirement Spec is a document written in a plain English language. It is a comprehensive manuscript primed by the punter corporation, using their computing information system which involves the complete synopsis of the computing system under consideration. It also contains the functional requirements and NFR's of the web based cloud computing systems, its actor/entity interfaces, control constrictions, and the archetypes etc. On the whole this SRS is a assortment of business requirements of diverse clustered stakeholders (end users of the computing organizations). The cloud computing service SRS is a set of requirements compilation of diverse clustered end users of the corporations. The SRS encloses ingredient branch of the computing system corporation mission charter. These pilots to the official launch of the computing system service software development of projects control the niceties a propos the outlay, calendar and the mission to be developed. It is jointly prepared and blended by client and the organizations. The SRS contains number of synonyms moreover; each individual code the requirement naming the items with context specific names. Thus SRS contains names such that each name coded by different users will be with different meaning such name or name phases as termed as heteronym.

Input: SRS

Output: Moulded Statements, statements with statement numbers.

i) Read the SRS in sequence if the assertion/avowal of the decree are compounded with ‘a’, ‘and’ and ‘or’ then next decompose merged complex decrees into uncomplicated multiple decrees.

ii) Translate reflexive influence to dynamic voice so as to renovate supporting auxiliary supplementary and uncongenial verbs into analogous parallel intransitive and transitive verbal representation

iii) Consign successively following numbers for each and every statement assorted.

3.1. Procedure for Moulding the SRS

SRS is a document includes FR, NFR's, defining overview of the computing systems, described in plain English language, also it contain actor interfaces, restrictions, POC's, prototypes etc. It contains the project charter, with cost, schedule, and all other details of the computing system. In many occasions it is multilaterally prepared and blended by service consumer and producer organizations.
Input: Software requirement spec
Output: cast statements, statements with numbered.

1. Scan and read the sequentially numbered statements, compounded with 'and' 'or' and later decompose the composite statements into simple tokenised multiple statements.

2. Convert the passive to active sentences voices, in order to transform the auxiliary and impersonal verbs, with intransitive to transitive verbs.

3. Itemize the consecutively succeeding statements.

Guidelines to design referenced- defined table

Referenced attributes on realization of the proclamation avowal if the assessment residue unaltered. Define the definite quality traits on the apprehension of the statement if the value remains altered.
Input: SRS statements with statement number.
Output: Referenced attributes, Defined attributes

Reorganize the statements such that the statements containing the defined attributes precede the statements containing those referenced attributes. This has to be represented in the tabular form as indicated below.

Table 1. Reference Attributes

Statement number	Referenced Attributes	Defined attributes

3.2. Control Flow Graph

Procedure to design control flow table (CFT)

The English language has million word languages with lot of flexibility. The SRS is documented in English language. Normally this SRS is collection of requirements of different end users of the organization. These different requirements are coded in English language with the cultural flexibility. Thus the document includes different words with the same meaning. The collection of these words is termed as synonyms. The SRS contains number of synonymies moreover each individual code the requirement naming the items with context specific names. Thus SRS contains names such that each name coded by different users will be with different meaning such names or name phrase is termed as heterogeneous heteronyms. There is a necessary to reorganize entire documents such that the attributes should be referenced only after defined. So there is a need for control flow graph.

(CFG): CFG is a directed rapt grid graphical G(V,E) symbolize everywhere v ∊ V, anywhere v is the contention or crowd together bunch of chronological statement of the program with one source of the program with one source vertex (in degree zero) with one or more destination vertices(out degree zero) and $e \in E$ is the directed edge connecting v_i to v_j.

Set up the logic based commonsensical succession cycle of the SRS statements that is symbolized in the outline of (CFT)
Input: Ref-Def table
Output: CFT

Procedure: i. As and when the proclamation is interpreted, sort the QOS attributes in the lexicographical order as shown in table.

Table 2. Intermediate table

Attributes in lexicographic order	Referenced number	Defined number

ii. Examine and interpret referenced-defined entries from the listed ref-def table.

iii. Create a 3 column as shown above table enter the assertion avowal item wise number in the right horizontal matrix rows of QOS traits at the accurate/precisely referenced or distinct vertical columns in matrix pilaster.

iv. If the defined entry follows the referenced entry, then delete defined entry and make the referenced number entry in intermediate table.

v. If the referenced entry follows the defined entry, then delete referenced entry and make the defined number entry in intermediate table.

vi. Sort the entries in the CFG with the jump1 or fly 1 as the primary key and begin as the secondary derived key from below table

Table 3. Start-jump table

Start	Jump1

vii. Enter the definite well defined and crystal clear proclamation with itemized number in the beginning of the matrix vertical column and the allusion proclamation number in the jump of the vertical column in table above

viii. If the defined entry does not follows the referenced entry or the referenced statement/entry does not follow the defined entry then it is left unconsidered.

ix. If the final CFG, create table with 4 columns with the start, end, jump 1 and jump 2 as shown in table below.

Table 4. Control flow table

Start	End	Jump1	Jump2

x. In the final CFG the first primary column beginning include the proclamation of the start/launch feature/traits.

xi. In the ending CFG in the table above the second column includes the last statement quantized number.

xii. Inside the final CFG the 3rd and 4th columns contain the jump and alternative jump statement entries indicated by the sentence number of the referenced and defined columns.

xiii. Maintain the modus operandi formula till the final proclamation avowal of referenced or defined in table.

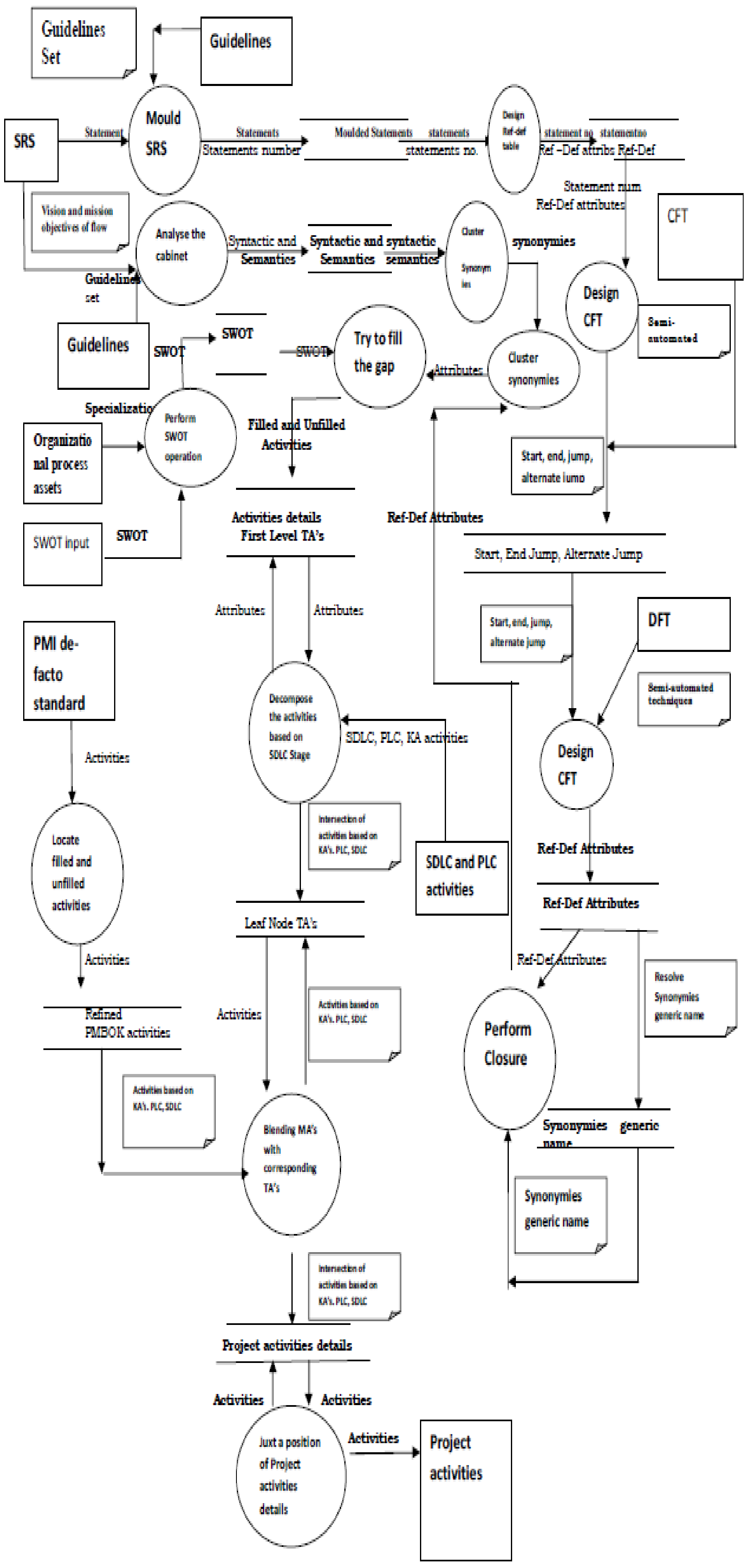

Figure 4. Data flow diagram trough the CCS PMA and PA

3.3. Data Flow Graph (DFG)

Procedure to design DFT: We recognize the ref-def table in the control (ruling) stream order to form data flow table. DFG is a concentrating directed graph G(V,E) clinch the deposit proviso vertices v ∊ V, such that where v_i represent referenced and defined attributes of the statement I and $e \in E$ represents an edge with statement at the head of the arrow data dependent on statement at the tail of the arrow. In the intervening statement the definition of i is preserved.
Input: ref-def table, CFT.
Output: Reorganized ref-def table in the control flow order (DFT).

1. Steps for identifying DFT in control flow order.
2. Start with the entry number in the first column of the CFT refers entries from table above.
3. In each statement identify referenced and defined attributes in the appropriate column refer entries from table above.
4. Continue this procedure till the statement number referenced in the 2nd column of the CFT in the same row refers entries from table above
5. Then start with the statement number in the 3rd column till it returns to the next statement. After the 2nd column statement number till it reaches the end of the program refers entries from table above.
6. Repeat this procedure for all the entries in CFT.
7. Reorganize the ref-def table in the control stream bid to form data flow table as indicated in the table below

Table 5. Data flow table

Statement number	Referenced attributes	Defined attributes

3.3.1. Procedure to Resolve Synonymies

The human always think in context specific, because of this we have flexibility in the English language different people use the same work in different meaning called synonymies.
Input: DFT.
Output: Synonymies in tabular form.
1. Read referenced attributes in tabular form.
2 Search for similar attributes entry in the DFT in above table.
3. Make appropriate entry in the synonymies referenced intermediate table as indicated in the table below.
4. Make the entry containing homogeneous attributes.
5. Continue this procedure till the end.
6. Repeat this procedure for next unmarked entry till all the unmarked entries are marked.

Table 6. Synonymies Referenced Intermediate table

Statement number	Referenced Attribute set in Lexicographic order	Statement number containing similar referenced attributes set

7. Repeat the steps 1 to 6 swapping defined and referenced attributes of DFT and make the entries in synonymies defined intermediate table below

Table 7. Synonymies defined Intermediate table

Statement number	Defined Attribute set in Lexicographic order	Statement number containing similar defined attributes set

3.3. Abstraction of Technical Activities

3.3.1. Procedure for Abstraction of Technical Activities

Input: SRS objectives list of prototypes, CFT, DFT.
Output: Technical Activities.
1. Identify noun and noun phrases from each objectives list them in the table below

Table 8. Noun-Noun Phrase table

Statement number of objectives	Noun	Noun Phrases

2. If the noun or noun phrases is not person or place identify the person or place involved in the entity (noun or noun phrase) and then store them in semiotic table shown below.

Table 9. Semiotic table

Involved syntactic	Semantics	Representing the pragmatic parts

3. Identify for each noun the synonymies (from the dictionary context) for each noun or noun phrases and then store them in synonymies table as indicated below

Table 10. Synonymies Referenced Intermediate table

Statement number	Referenced Attribute set in Lexicographic order	Statement number containing similar referenced attributes set

Table 11. Synonymies defined Intermediate table

Statement number	Defined Attribute set in Lexicographic order	Statement number containing similar defined attributes set

4. Consider each synonymy, search for different element of SRS, If present identify the referenced element corresponding to the synonymies element.
5. Identify the closure vector with known initial item and final known defined items as indicated below in the fig below

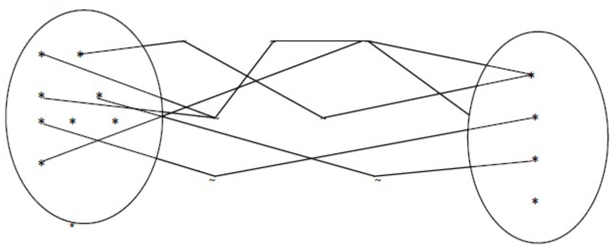

Figure 5. Shows the closure operation

6. Search for the purpose of vector element in the list of the prototype and make the entries in the table below

Table 12. Prototype Table

Prototype name	Initial element	Intermediate element	Final element

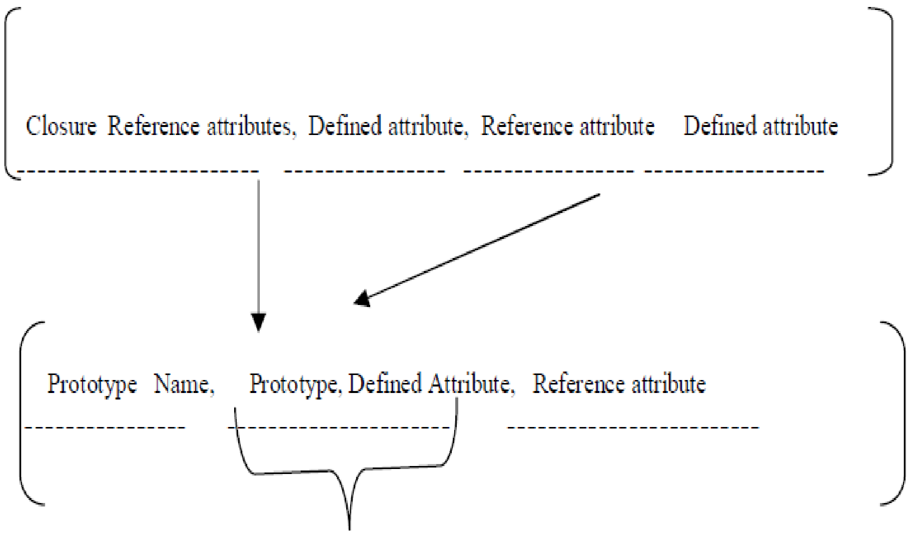
Closure Reference attributes, Defined attribute, Reference attribute Defined attribute
Prototype Name, Prototype, Defined Attribute, Reference attribute

Figure 6. Mapping closure operation

From one prototype we search for the prototype from input to output. Initially start from the attribute and read till i+m attribute. For strength start from i+m attributes till it reaches i+m in the positive direction on reaching the output. For weakness start from 1+1 attribute if it reaches the i-1 in the negative direction on reaching the output. The SWOT analyses and identified as follows:

SWOT analyses (Strength, Weakness, Opportunities, threats)

Strength: The parameters of the input features which facilitates to move the input syntactic in the following direction to adjective syntactic.

Weakness: The Parameters of the input features that obstruct the facilitating that features towards meeting the objectives.

Opportunities and Threats: these are analogical to strength and weakness of the system available in the computing ecosystem.

- Identify the strength of synonymies of input with respect to the organizational process assets and the SRS by the features facilitating the activities. That moves the subset of each closure path in the forwarding direction towards meeting the objective synonymies as shown in the figure below.
- Identify the weakness of synonymies in input with respect to the organizational process assets and the SRS by the features abstracting the activities that move the subset of each closure path in the forwarding direction to meet the objectives synonymies as shown in the figure below.
- Identify the threats from the enterprise business environment and ecosystem factors that is abstracting leveraging of the input synonymies in moving the subset of each closure path in the forward direction to meet objective synonymies as shown below in the figure.

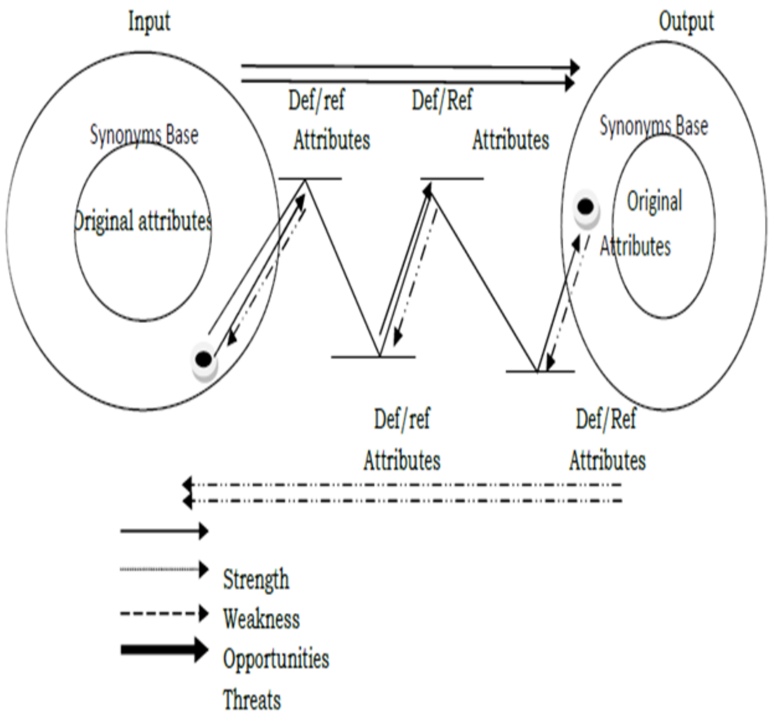

Figure 7. SWOT Analysis

3.3.2. Managerial Activities

Software Project Management

The first level PA's serves as basis for determining remaining PA's and they are classified according into PLC phases, SDLC stages, KA's. Software Development life cycle stages: In this work we considered 7 SDLC stages viz. requirements analysis, High level design, Lower level design, Unit code, integrated code, unit testing, integrated testing, acceptance testing, implementation, Delivering, software maintenance. Project life cycle Phases: Each project should follow 5 PLC phases viz. commencing Phase, scheduling/forecasting phase, Implementation and executing phase, scrutinize, supervising and domineering, and closing phase.

Project Management knowledge areas: Based on task the project management are classified into different knowledge areas for software development projects, the activities are clustered in 10 KA's viz. Cloud computing system Project service Integration and administration, cloud computing system Project service Scope and dimension management, CCS Project service Time schedule management, CCS Project service outlay administration, CCS mission QOS value administration, CCS mission service HR and computing resource management, CCS Project service communication management, CCS Project Service risk management, CCS Project. Service procurement management, CCS Project service configuration management. The cluster of camouflaged activities forms the first level project technological activities

$$\text{Technical Activity} = \text{SDLC Phase} \cap \text{PLC stage} \cap \text{KA}$$

Differentiation linking CCS Software project and other computing system projects

Table 13. Cloud projects and other projects

CCS Software Projects	Other Projects
Syntactic => Words, prepositions, attributes	Syntactic => each part

Table 14. Showing syntactic and Semantic behaviour

Syntactic => Words, prepositions, attributes	Syntactic => each part
Semantic => statements	Semantic => combination of elementary parts
Pragmatics => module	Pragmatics => conjoining of parts serving a purpose
Behavior => The serving purpose of the product /service/result	Behavior => The serving purpose of the product /service/result
Behavior $=\bigcup_{i=1}^{n} pragmatics$ where n varies for some purpose. There is no scope for change in behaviours	Behavior $=\bigcup_{i=1}^{n} pragmatics$ where n will be fixed. There is no scope for change in behaviours
Components are not fixed	Component is fixed
Support configuration management using Dev-ops	Do not Support configuration management using Dev-ops
It gives scope for modification, since it as to obey the open close of good software principle. Thus the behaviour is linked to the list of pragmatics	There is no scope for modification, thus behaviour is array of pragmatics
Linked list varies with the human skill dependent	Array is not human skill dependent
The elements cannot be replaced. n varies it depends on the human dependent, it is difficult to identify the node in the linked list and it is difficult to reassemble	It is an element of the array does not work it can be replaced,
It is intangible	It is tangible
It cannot be easily detected and replaced, we take the support of configuration and management	Easily detected and replaced.

Table 15. Showing various steps of Invisibility factors

Invisibility

$n \approx$ developers skill	$n \approx$ Product/service/result
Pragmatics = f(syntactics, semantics) where f varies with human skills	Pragmatics = f(syntactics, semantics) where f depends on product, service or results.
Pragmatics is not measurable	Pragmatics is measurable

Table 16. Showing Various Human resources factor

Human Resources

Pragmatics = f(syntactics, semantics) where f varies with human skills	Pragmatics = f(syntactics, semantics) where f is fixed

Table 17. Showing various steps of conformity factors

Conformity

Pragmatics = f(syntactics, semantics) where f varies with human skills	Pragmatics = f(syntactics, semantics) where f is fixed
Pragmatics in software can be organized in number of ways	Pragmatics can be organized in a single way
Pragmatics in not conformal	Pragmatics is conformed

Table 18. Showing various steps of schedule factors

Schedule

The software activities depend on the results of the previous software	Some parts can be prepared concurrently
We have to use depth first search algorithm	No algorithm is required
Order cannot be decided previously	Order can be decided previously

Table 19. Showing various steps of various factors

Tools and techniques used vary	Tools and techniques used is fixed
We don't know the infrastructure	We know the infrastructure
Tools and techniques developed vary	Tools and techniques developed is fixed
Input varies	Input is fixed
Cost estimation varies	Cost estimation is fixed
Schedule increases risk also increases Probability α 1/risk	Schedule is fixed hence risk is known
Organization process set varies	Organization processes set are fixed
Enterprise environment factor varies	Enterprise environment factor fixed

3.2. Hazards of Developing Technical Activities Based on the Managerial Activities

The purpose of the administrative activities is to augment the QOS of the technological activities (TA's) should be in place. Though the technical activities may depend on the CCS project service input and the aims of the project but there is at-least a possibility to design and develop a tactics strategy to obtain technological actions in the absence of such methodology the software developers are blindly following the **(MA's)** to conceptual the TA's.

Managerial activities need to quintuplet of quality parameter value of the TA's to augment the QOS values. The harvest of the TA's contains different updating version of the quality values. At different values of time the managerial activities are the activities those enhance the quality values from one level to the next higher level values in the consecutive order. Managerial activities are defined in 2-D plain comprising project life cycle phase's axis and knowledge axis. Abstracting technical activities from the managerial activities is analogous to transform 2-D activities into 1-D activities. Which is impossible, in case if both the dimensional values vary in the order pair of 2-D value!. Either the first or the second value should be content then only the technical activities may be developed if we achieve dimensional equipollency

4. Continuous Uninterrupted CCS Service Software Security Development Model

Continuous cloud computing system service software security progressive framework is to mitigate computing system security risks, in the initial stages at the paramount. In Cloud service consuming corporations and service producing ventures fabricate computing system software at faster pace on a clock cycle speed, due to marketing time is crucial for their business existence. At present software as a service speediness means computing business. Traditional security measures do not frequently perk up with the modern velocity of computing system software service delivery. Safety measures requirements to be assembled into the computing service software. Even prior to coding computing systems software, security is bolted to the computing system with its fabrication of the software. In general the designed security is breached separately by folks with malevolent purpose. This research case study investigation work confers a multilevel computing system Software Security ripeness model that is a synthesized with union of population, processes, plan, procedures and tools that enables applications and the data they process are secure.

Whilst the construction team is concentrating on sustainable deliverance, virtual cyber hackers or attackers labelled criminals are engaged on discovering flaws, errors, mistakes, vulnerabilities/loopholes and keep exploiting them for their benefit and gain to keep out the service vendor from offering the service. Defending application service on the cloud based Internet is a 24/7 trade. Whatever things needs to facilitate the customer is coupled to the Internet is an objective target of cyber abuse these times. The impacting of a defence episode can only be calculated precisely by measuring in terms of trouncing of assets, thrashing of reputation and share, disappointing burden on profits, annoyance of functions, or even insolvency in terms of economic failure and service disruption. The degree of cyber assaults has also broadened and amplified along with technological innovations and type of computing to embrace any size association and business firms across the industry. Whereas the firms mainly organizations ponders principally on computing system functionality, profit making and getting ROI and speed. Security is not a nucleus competency of computing systems coder. This in succession means that software, ahead finishing point, surround several vulnerabilities that masquerade threats signifying real-world risks with respect to web computing amounted to data or information breaches in 2016 in a survey report. Close at hand used to be an occasion when a plain firewall and anti-malware/antivirus software solution were all that was obligatory to maintain appliance secure on the Internet web which is insufficient for present security breaches. Fortification regularly commences by securing by securing the cloud computing communications infra as a service, then with CCS podium for computing, and application developed over using both computing services apart with 360^0 security round the clock with geo fencing, and also by continuous refactoring using Sec-Dev-Ops processes. Primarily, nonetheless weak computing software leaves open doors for cyber attacks on critical data and computing info systems.

Although violations, infringes and security episodes have transpire an undividable ingredient of today's commerce, crops up because of frequently vulnerable simple or complex computing system software architecture design and coding flaws. It terms for a united approach pooling and innovating on engineering best practices, to computing software security at a tactical level. To administer, implement, employ and execute to gauge the advancement of the uninterrupted computing system software security (sec-dev-ops) journey. An amalgamation of dynamic, static, semi dynamic, semi static and hybrid appliance automated scanning tools pooled with coder/programmer education, standards, guidelines, and security acumen progress security in the SDLC. Continuous Delivery status impediments and delivery process main principles are: Automation process completely in cloud environment. If something wrong detected, it requires to do refactoring very often, on the definition of done applies for liveliness, with built in quality, implementing the improvements and innovations continuously with entire stakeholders accountable for the output. The benefit of security continuous refactoring includes shorter time to market and implementation with improved better quality of security implementation. Improved secured business confidence, with faster output, implementation at lowered operational costs. To comprehend these needs, the automated construction, verification and validation, operation, and service provisioning course of action plays a vital role ROI with room for improvement.

4.1. Continuous Cloud Computing System Service Software Security

Model the five stage development model focal point on the vital elements of an institute that desires to be reactive to the marketplace changes. At the equivalent time focal point on less everyday progression that measured slows down the velocity of computing service software delivery. The primary essentials comprise tutoring, physical authentication, and automatic verification via dynamic and static appliance hybrid security testing, computing build integrations testing, security implementation requests, software functioning environment, and event response. New facades of computing system security are pioneered with succession throughout diverse intensities. With progression at the entire level, around is powerful hub on automation, and also manual confirmation verification, screen and continuous supervising helps in thwarting vulnerabilities. Various stages includes as follows

4.1.1 Fundamental (stage 1) for any corporation, the stage is likely lowest to be in relegate. Individuals are well-informed about essentials of information security, what ought to be prepared and what must not be done, for clearing up the system from the terrorization of the present virtual computing web cyber world and the real globe risks that arrive from the unsecured computing system. The stakeholders are educated for cyber defence hygiene. The Quality Analysts (QA's) concerned in computing system software development investigate for border line geo fencing cases everywhere they begin assessment with negative as regards what may perhaps go erroneous. There is no computerization defence testing present at this stage. The fundamental defence rules that are afforded by the computing system framework trader as a measurement of the paradigm quality checks that are made as a ingredient of the active SDLC process.

4.1.2 Investigative phase (stage 2) Being at the investigative probing level is a high-class position to commence. Software safety scheme can be designed. The stage starts with recognizing and grading data and application types that are at hand in the venture. From there, open with what matters on the whole. Recognize the crest list of appliances or locales that should to be protected and begin with preliminary application safety measuring guidance. Educate coders how to hack appliance so they can avoid them from conquest of hack in the authentic production environment. Physically analyze vulnerabilities for the vital vicinities. Focus on automating vulnerability investigation for the appliance parts uncovered via the GUI by repeated crawling methods. Computerize the course of scanning the crucial appliance resource code recurrently to ascertain and unearth vulnerabilities at the root architectural design and source code stage. Amalgamate the semi or full Dynamic App safety Testing (DAST) and semi or full Stagnant App Safety Testing (SAST) and/or hybridized technique of mutually DAST and SAST grades as a measurement of the computing system build productivity so coders can glance at them and identify with the varieties of threats that the appliance are up against.

4.1.3 Protected SOFTWARE incorporation (stage 3) the protected Software incorporation set in motion with a computerization and automated strategy as one of the large vital focus area. Train the coders with the profundity of secure programming models through workshops. Assist them identify with crucial application controls like approval, cryptographic coding, encoding, decoding, inaccuracy management, and input substantiation, information encoding and managing alike. Perform episodic Weakness investigation and infiltration and breach Testing (Web Application Service Vulnerability Apps and Penetration Testing (WASVAPT)) on the application. Carry out cipher security code reassess. Describe programming guidelines and build them as guiding principles for peer to peer code checking processes prior to the code gets chequered into the source version control storehouse repo. Allow the coding principles be definite based on Industry guidelines like the OWASP Top 10, OWASP ASVS, cloud security alliance, NIST, SANS Top 25, guidelines identical. Automate (DAST) using standard tools. Facilitate the automated tools to memorize verification, validation and conference session, and end user/stakeholders details and agenda based automated checks to run numerous epochs a day. Run SAST at each coder commit to the source code repos. Fine-tune the automated tools to provide them as a good deal of context as to lessen the quantity of spurious positives. Split the build for crucial key security findings. Mould the appliance for detection of terrorization and hazard agents. Launch architectural design level mitigations and articulate security requirements since recognized risks prior to the code are in reality written. Capture benefit of Continuous Integration atmosphere to discover and alleviate threats early on and frequently.

4.1.4 Safe and Sound SOFTWARE Deliverance (stage 4) at the protected Software deliverance stage, it is essential to encompass tradition and automated tools to aid us mitigate terrorization at a sooner pace. Self-protective architectural design and encoding are an extremely vital part of quicker release cycles. Perform systematically task specific appliance security education for all stakeholders involved in the software deliverance enterprise. Recognize and keep up the transforms that be prepared all through repeated discharge cycles. Perform WASVAPT on the functioning milieu. Preserve security stipulations for running atmosphere. Investigate the potential of function firewall to stop molest in bona-fide time. Mechanize DAST and SAST for the new transform that has been devoted while the past discharges rotations. As swiftly as the origin code is committed to the spring controlled depot, elicit the tests and split the build according to security policies. When the source code is committed is broken down, alert stakeholders a-propos the unbalanced codebase. Amalgamate effort tracking or virus, defect tracking systems to the trial results. Spot software conformity needs early on adequate to articulate them as preset security requirements. Preserve a catalogue of suggested software frameworks that are accepted to be used for software maturity. Make certain an event reaction progression is clear and in consign to react to molest as the appliance is resided in real production environment, install repeatedly and quicker, steadily.

4.1.5 Protected software release and supervising (stage 5): towards be revealed a Software safety champion, mechanization require to be pooled with continuous supervising and analytics. Consent role-based appliance defence inspection as a measurement of worker on-boarding modus operandi. Episodically appraisal functioning milieus with engineering yardstick benchmarks and watch the functioning atmosphere for base lining the constitution transforms. Consider measures to protect all the associated schemes and backend services together with third party sellers providing computing system as a service or services comparable. Security serviceable needs to be programmed, for instance constitutional rights precise function of an appliance. Prop up an ethnicity of inscription security regression tests for software defense fixes that are concerned. Recognize incessant enhancement areas, and pioneer secure mechanism at the software framework level. Amalgamate secure source code investigation tools at the coder IDE for quicker response. Consolidate result from copious security tools and continuously correspond the security potency of appliance. Dwell in progressive knowledge like bona fide Real time appliance security defence (RASD) to discover and react to coercion in real-time.

Safety measures are an endless scuffle that desires indeed to fight with intelligence and vigil. Even though executing the security in Software might resonance irresistible in the commencement, pilot stale with a software safety measures. Maturity program offer a design of the modern eminence and a boulevard for enhancement. Security computerization does not purge the need to bond manual re-examine. It assists focal point on vicinity where equipment has precincts and anywhere individual proficiency conveys mainly assistance. Festering ad-infinitum delivering software at rapidity does not inevitably signify more weakness convey at a quicker pace, but more Agile safety measures and resiliency built within the application accurate from the induction, persistently.

Geo -lattice (geo-fencing)

Geo-fencing (geo-fencing) is a trait in a software code that uses at the widespread (positional) location scheme (GPS) or radio frequency identification (RFID) to describe, label geological borders (precincts). Geo-fencing permit an supervisor to set up elicit so whilst a gadget pierce (or outlet) the precincts cleared by the supervisor, an alert is alarmed. Numerous geo-fencing appliances integrate Google Earth, permit bureaucrat to identify borders on zenith of a satellite vision of an unambiguous ecological area. Other appliance defines borders by longitude and scope or in the course of user -fashioned and Web-based maps. Geo-fence virtual barricades can be dynamic or reactive/flaccid. Dynamic geo-fences necessitate an end user to opt -into to locality services/forces and an itinerant app to be unbolt. Inert geo-fences are constantly on, as they rely on Wi-Fi and cellular statistics as a substitute of GPS or RFID and operate in the background.

The machinery has several realistic uses, include:

Table 20. Showing various geo fencing factors

Uses	**Example**
buzz management	An admirable incident can use geo-fencing to build a provisional no-fly sector that thwart whirr from voyage a definite outer limits.
task force management	Geo-fencing can vigilant a reporter when a data traffic driver code flipside execution route.
Stakeholder and resource entity management	A workforce entity smart card will post an observant alarming message to defence if the worker try to penetrate illicit, geo-fenced locale vicinity.
conformity management	Set of connections logs can support geo-fence voyage crossing to manuscript the accurate use of automated tools and smart devices connected to network and their conformity with recognized guiding principles.
Advertising and marketing management	A petite commerce can transcript an opt-in patron a voucher code when the consumers Smartphone come in a definite physical area.
Resource management	A system administrator can locate and alerts so when an authorized smart device is stolen or foliage the boundary which is supposed to be in place, the admin can monitor the device's locality and padlock it to avoid it from being worn and utilized illegally
regulation enforcement	An ankle wristlet can on the alert establishment if an entity under residence seize cross or leave the location.
residence automation	When the residence proprietor Smartphone stolen or leaves the home geo-fenced outer limits or boundary.

5. Conclusion

Either core activities or technical activities are developed from the managerial activities. The task of which is to enhance the quality of the core activities. This is derision of foppishness. In this work this has been eliminated by our new methodology for the design and development of core activities from the input features and the objectives. The synonymy issues have been amicably resolved. The technical activities and managerial activities are dimensionally different. In our work we have developed methodology for the attainment of dimensional equipollency. Thus through mathematical empiricism we have optimized blending of two types of activities to achieve quintuplet of quality parameter viz. correctness, completeness, robustness efficacious efficiency. Most of the tools and techniques of managerial activities depicted by PMI's (PMBOK) are left to expert judgement are clairvoyant study shows that, the reticulation of just positioned interleaved technical and managerial activities is not known. So we attempted to develop the reticulation of just positioned interleaved technical activities and managerial activities which provide different updated versions of attributes to managerial activities in these cases the process of elimination of "Expert judgement". We also presented the effect of scope creep in project management using various case-studies, and illustrations

References

[1] Project management body of knowledge, PMBOK 4th and 5th edition, published by Project management Institute, USA (Reviewed once in 4 years).

[2] Software project management a real world guide to success Joel henry pearson education inc 2004.

[3] Software project management a concise study by Kelkar S A PHI learning private limited New delhi 2nd edition 2011.

[4] SM Handigund, BP Deepak, "Reverse engineering of legacy cobol systems", Ph. D. thesis Indian Institute of Technology Bombay, 2001.

[5] Software project management a unified framework by Walker Royce Pearson education inc.

[6] Shivanad M. Handigund, D. B. Phatak in proceedings of "Reengineering Methodology for Legacy COBOL Systems" has been published in proceedings of International conference on Information Technology held during August 01-04, 2000 at Bangkok, Thailand.

[7] Shivanand M. Handigund, Shakuntala Sajjanar, B. N. Arunakumari: "Resuscitation of syllogism within unified modeling language levels through the renovation of object diagram". ICACCI 2015: 2397-2401.

[8] "Resuscitation of syllogism within unified modeling language levels through the renovation of object diagram". ICACCI 2015: 2397-2401.

[9] Shivanand M. Handigund, Siddappa G. Makanur, M. Sreenivasa Rao., "Integration of Object Oriented Host Program with Network", DBMS. SCSE 2015: 178-185

[10] A. M. Shivaram, Shivanand M. Handigund. "An Ameliorated Methodology for the Abstraction of Object Oriented Features from Software Requirements Specification". SCSE 2015: 274-281

[11] Shivanand M. Handigund, B. N. Arunakumari: "An Ameliorated Methodology to Abstract Object Oriented Features from Programming System". SCSE 2015: 470-477.

[12] Shivanand M. Handigund, Shivaram A. M., B. N. Arunakumari., "An ameliorated methodology to establish the analogy between business process perspective views and UML diagrams. ICACCI 2014: 231-23".

[13] Shivanand M. Handigund, B. N. Arunakumari. "An ameliorated methodology for the identification of project activities of software development projects". ICACCI 2013: 172-177

[14] Shivanand M. Handigund, Bhat Shweta. "Automated Methodologies for the Design of Flow Diagrams for Development and Maintenance Activities". CETS 2010: 93-104

[15] Ajeet A. Chikkamannur, Shivanand M. Handigund. "An ameliorated methodology to design normalized relations". AICCSA 2009: 861-864.

[16] Shivanand M. Handigund, S R Nagalakshmi. "An ameliorated methodology for the design of panoptic work process flow diagram", Conference: 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) May 2016.

[17] Shivanand M. Handigund, S. B. Kshama, N. Ranjana. "An ameliorated methodology for design and development of a work process diagram to be incorporated in UML." ICACCI 2013: 184-190.

[18] Shivanand M. Handigund, Rajkumar N. Kulkarni. "An Ameliorated Methodology for the design of Object Structures from legacy 'C' Program", 2010 by IJCA Journal.

[19] Shivanand M. Handigund ; Shivaram A. M. ; Arunakumari B. N. "An ameliorated methodology to establish the analogy between business process perspective views and UML diagrams", (ICACCI, 2014 International Conference on Advances in Computing, Communications and Informatics, 24-27 Sept. 2014, IEEE.

[20] S. M. Handigund and R. N. Kulkarani. "Abstraction of structural and behavioural components from legacy C program" in the proceedings of 2nd International Conference on Advanced Computing and Communication Technologies (ICACCT-2007) November 03-04, 2007 at Paniput, Haryana.

[21] Shivanand M. Handigund; A. M. Shivaram. "An Ameliorated Methodology for the Abstraction of Usecases from software requirements specification", IEEE, Science and Information Conference (SAI), 2013, 7-9 Oct. 2013.

[22] Shivanand M. Handigund; H. B. Kavitha; A. Aphoorva, "An automated methodology for the design of usecase package diagram", 2012 IEEE-World Congress on Information and Communication Technologies, Year: 2012, Pages: 309-314.

[23] Shivanand M. Handigund; Shweta Bhat. "An ameliorated methodology for the abstraction of object class structures for an information system", 2010 International Conference on Computer Applications and Industrial Electronics (ICCAIE 2010), December 5-7, 2010, Kuala Lumpur, Malaysia, IEEE, pp-362-367.

[24] Shivanand M. Handigund; Nagalakshmi S. R.. "An Ameliorated Methodology for the Design of Panoptic Usecase Flow Diagram to Constellate UML's Usecase and Sequence/Communication Diagrams", 2014,IEEE- Fourth International Conference on Advances in Computing and Communications, Year: 2014, Pages: 153-156.

[25] S. M. Handigund; C. Pushpa. "An ameliorated methodology for the design of Product Data Flow Diagram", 2012 World Congress on Information and Communication Technologies (WICT), 2012, IEEE, 30 Oct.-2 Nov. 2012.

[26] S. M. Handigund and R. N. Kulkarani, "an Ameliorated Methodology for the Abstraction and Minimization of Functional Dependencies of legacy 'C' Program Elements", International Journal o f Computer Applications.

[27] Ajeet A. Chikkamannur, Shivanand M. Handigund ," A Mediocre Approach to Syndicate the Attributes for a Class or Relation", International Journal of Software Engineering and Its Applications, Vol. 5 No. 4, October, 2011

[28] Shivanand M Handigund, Syed Naimatullah Hussain, "Design of an Ameliorated Methodology for the Abstraction of Usable Components of Object Oriented Paradigm from the Software Requirement Specification (SRS)", International J. of Recent Trends in Engineering and Technology, Vol. 3, No. 2, May 2010.

[29] Shivanand M. Handigund, "A Suigeneris Automated Software Testing Methodology", Procedia Computer Science.

[30] Volume 62, 2015, Pages 21-22, Elsevier 2015 International Conference on Soft Computing and Software Engineering (SCSE 2015).

[31] Shivanand M Handigund, Shakuntala Sajjanar, "An Ameliorated Methodology for the Determination of Shortest Path in Single Source Multi Destination Graph", Conference Paper · January 2015, Conference: Department of Information Science and Technology.

[32] Shivanand M Handigund, Priya srinivasa, “An Ameliorated Methodology for the Design and Development of Software Project Activities,” Conference paper, Jan 2015.

[33] S. M. Handigund tutorial on “Unravelling The Hidden Treasures Of Research Avenues In Object-Oriented Technology” Worldcomp'11 at Las Vegas, Nevada, USA on July 18-21, 2011.

[34] S. M. Handigund, Shivaram A M and Apoorva “An Ameliorated Methodology for the Abstraction of Use Cases from SRS”, international journal on Global Perspective on Engineering Management (GPEM) published by the World Academic Publishing Co., Limited, Hong Kong.

[35] SM Handigund. “An Ameliorated Methodology for the Abstraction of Use cases from Software Requirements Specification” has been published in IEEE technically sponsored international conference Science and Technology (SAI-2013), London, UK to be held on October 7-9.

[36] Shivanand M. Handigund, Kshama S. B., Ranjana N. “An Ameliorated Methodology for Design and Development of a Work Process Diagram to be incorporated in UML”, 2013. International Conference on Advances in Computing, Communicationsand Informatics (ICACCI). Pp-184-190, 22-25 Aug. 2013.

[37] Pankaj Jalote, An Integrated Approach to Software Engineering, 3 Edition, Narosa Publishing House, New Delhi, 2008.

[38] Shivanand M. Handigund, Reverse Engineering of Legacy COBOL systems, Ph.D. thesis, Indian Institute of Technology Bombay, 2001.

[39] Software Engineering: Ian Sommerville, 8th Edition, Pearson Education, 2007.

[40] John Atkins and Jon Beck. A survey on program slicing for software engineering. In Technical Report, Dept. Statistics and Comp. Science, West Virginia University, West Virginia, April 1993.

[41] Lioyd D. Fosdic Leon J. Osterweil. Data flow analysis in software reliability. SIGPLAN Notices, 8(3), Sept.1976.

[42] Jaco de Bakker and Erik de Vink. Control flow semantics. The MIT Press, Cambridge, Massachusetts, 1996.

[43] Michael Wolfe. High Performance Compilers for parallel Computing. Addison- Wesley Publishing Company, Redwood City, 1995.

[44] Werner Kluge. The organization o reduction, data flow and Control flow. The MIT Press, Cambridge, Massachusetts, 1992.

[45] Lawrence Markosian et. Al. Using an enabling technology to reengineer legacy systems. Communication of the ACM, 37(5), May 1994.

[46] Frank Tip. Generation of Program Analysis. Ph.D. thesis, Amsterdam University, 1995.

[47] Mark Weiser. Program slicing, in IEEE International Conference on Software maintenance, 1981.

[48] Mark Weiser. Program slicing. IEEE transaction on Software Engineering, 1984.

[49] Larry J. Morell, Lionel E. Deimel, “Unit Analysis and Testing”, SEI Curriculum Module SEI-CM-9-2.0 June 1992.

[50] Agostino Cortesi Gilberto Fil´e (Eds.) Edited by G. Goos, J. Hartmanis and J. van Leeuwen, “Static Analysis”, Lecture Notes in Computer Science 1694, 6th International Symposium, SAS’99 Venice, Italy, September 22-24, 1999 Proceedings (Lecture notes in computer science ; Vol. 1694).

[51] Software project management by Pankaj Jalote, Peason Pubication, 2011.

[52] Software Project Management by Bob Hughes and Mike Cotterel, Tata McGrawhill, 5th edition, 2011.

[53] Frank Tip, Generation of Program analysis PhD thesis Amstredam University1995.