Decision Support System for Software Release Management

Sidra Anwar

ABSTRACT—Software release management process is a decision process and includes deciding on release item's contents planning, requirements priorities, change requests and development trail. To manage release processes effectively and efficiently, several algorithms and approaches have been proposed for each of the processes separately, to help software developers to develop and manage coming releases. The current study, however, proposes a new approach by developing a decision support system (DSS) to keep track of all release processes in software development life cycle.

A case study is performed to test this decision support system on the overall release development and management procedures of the product under study: "study on cloud". Furthermore, results are measured on the basis of health, quality and progress of the software code after implementing the concrete architecture of the decision support system for software release management. The study further assesses and shows the impact of the solution system on release managerial aspects.

Index Terms—Software Development Life Cycle, Software Release Management, Decision Support System, Study on Cloud

I Introduction

An ideal product is one which can be achieved with efficient project management and defects prevented and detected earlier within the product development life cycle [1]. The product development procedures have been manual to a specific degree [2] and the major problem in such a Lifecycle is that project shortcomings, changes or risks are usually discerned late in the cycle and thus, the controlling and curing the problems becomes expensive. It can be depicted as the issues and delays in procedures for decision making about the evolutions, implementations and management of coming releases of a software product [3, 1]. It is a key innovation for delivering the item to the client on time. The key achievement of any product item lies in how delicately the item is released to the client and how viably the assets are allotted and used to accomplish the required result [4].

Manuscript is submitted on July 24, 2019. This work was supervised in COMSATS Institute of Information Technology by Dr. Liaqat Ali Shah.
Sidra Anwar is currently working with the GC Women University Sialkot, Pakistan, phone: +923246110034; e-mail: engrsid.es@gmail.com

The study of software release management arose when coordination and change control became a problem in software projects. Software release management is the way towards deciding, controlling, developing, releasing and executing changes in an IT environment [5]. In IT Industry, The current trend is moving towards deliverable based components development and autonomously assembled released products [6]. Geographically distributed and large development teams increased the demand of set of tools to support the release management processes. While release built and management, certain issues are found, like updated versioning, automatic built support, multiplicity of concerns and artefacts repositories.

As for the Decision-making in Software Release Management (SPM), other than development and management issues, the success and failure of a software product depends on the decisions regarding change deployment, requirements prioritization [7] and the contents of coming releases. As for prioritization and contents planning, numerous manual and automated algorithms have been proposed to perform series of steps to generate possible decision outcomes [8]. Several approaches have been proposed for decision support and problem concerns, such as binary search tree for concerned requirements priorities [9], linear programming for software optimization [10] and genetic algorithms for planning releases. All those approaches and algorithms required to be enacted by humans manually or with the support of a computer program. There is no framework for management and incremental deployment of such solutions, monitoring and assessing their performance, and improving and optimizing them.

Early studies identified decisional management issues as a root cause of development problems such as costs overrun, late schedules, low quality products and customer dissatisfaction [11] so improving these management processes stipulate the improvement of whole software development processes in general. Many solutions have been proposed to remedy the managerial issues [12] but software industries

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

seldom adopt such solutions which require direct involvement of management in development processes.
To identify the issues facing software release management or decision making, one must determine first that what is expected of the management. Management must set goals and provide directions for the successful release development [13]. It must devise a plan and monitor the progress made towards this plan, e.g., operational analysis [14]. When problems arise, management must initiate impact assessment of the problem, search for potential solutions and their implementations. Furthermore, in lieu of continual evolution of the release item, management must develop a standard system to make strategic decisions to resolve problems regarding decision making, e.g. strategic analysis [2].

To put management inaction, one must take into account three managerial concerns [3]. First, proposed solutions to software engineering problems are partial or temporary as software systems continually evolve, which results in changes in the nature of identified problems [9].Second, problems of software engineering reflect the multiplicity of concerns that the management must face. Quality objectives, resource allocations, skills, cost and schedule constraints are among these concerns [15]. Such concerns are not independent of one another and management must make compromises in addressing them together. Third, software engineering still remains in its infancy with regards to empirical studies [16] because reported problems are scrutinized in the literature and counter-examples put forward, which results in the problem or its proposed solution becoming suspect.

Dealing with first problem, there should be an application that can always manage the evolutions and change requests in the phases of software engineering at any stage and update the whole system development life cycle with every implemented change [5]. Another factor to consider while implementing this solution is to decide which changes should be implemented and which should not [17].

Moving towards the second problem of multiplicity of concerns, there should be a support system for management to easily handle these concerns like resources, skills, and cost and schedule allocation [14] but not in classical method using the tabular representation, which is not even an efficient method. And there are also many management issues for whole team to visualize their roles and resources etc. [18]. So there should be a platform where whole work flow would be clear at run time even in distributed teams within no time with alert and notification. Again there will be little management issues regarding decision making that to whom and when to assign the resources and also to keep track of all the information.

Considering the third problem, it would be a good start to practically implement the proposed solutions in software development environment to analyze the problems to some extend so far and also an opportunity to test the possibility of their applicability [9]. So such (informational) decision support system is a proposed remedy which could enable management to independently validate the existence of such problems, assesses their impact, and evaluates the potential benefits of the proposed solutions in any phase of software development process.

An obstacle to handle these managerial concerns is the lack of supporting automated system. In many cases, despite promising tools and technologies, there is little data supporting managerial decisions needed in release management. There are several models for defect predictions [7] and to find acceptable solutions for such decisional issues [19], yet all of them have come under severe scrutiny [20, 5]. To deal with the problem, this research proposes to model and evaluate the problem to support decision-making in managerial issues in software release management independent of classic techniques. Furthermore, it proposes a system for monitoring, prediction, and analysis of these goals based on factual data, which in turn could support the decision-making process. The flexibility built into this framework provides a facility for continual improvement in software development manages the multiplicity of concerns and enables continual improvement of the product management in lieu of software evolution.

II METHODOLOGY

Software development life cycle (SDLC) is a procedure followed by organizations to configure, create and test their major software projects. Each phase produces deliverable stipulated by the subsequent stage in the life cycle. Requirements are converted into plans. Program is developed by the plan and subsequent to development and improvement stage; the testing verifies the deliverables against requirements and delivers to deploy further. Major changes occur at testing or development phase of this Life Cycle. This solution model will help to manage such software changes that turn into releases. Major change or release bundle will be adhered to this support system for software release management while minor operational or emergency changes are not entertained.

In software development life cycle, when problems arise, management must initiate impact assessment of the problem, search for potential solutions, their implementations and develop a standard system to make strategic decisions. An obstacle facing management is the lack of supporting automated system. To deal with the problem, a model is

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

proposed to automate the release processes in SDLC and to support decision-making in managerial issues in software release management. To find a suitable solution via this support system, follow the fig.1 for complete workflow.

A. Parallel Processes:

1. Pre. RFC Process (PT-2)

In the Role of release engineer or analyst in fig.2, as it's mentioned that after the creation and start of release tasks in A2 and A3, pre Request for change (RFC) process will be performed before passing change to Change advisory board. In pre. Request for change process, a new RFC will be created by a requester in SA-1. At next step of SA-2, the requester is needed to be identified to ensure his worth for request for change in smooth development procedures of the system. Furthermore, in SA-3, determine the type, scope, schedule and risks of the requested change on the system and prepare in the form of document plan at the step SA-4. And lastly, before passing the request to CAB, attach this supported document plan with the request as a help for better and quick decision in SA-5.

2. Post RFC Process (PT-3)

When a change for request is passed to Change Advisory Board (CAB) in fig. 3 where in A-14, post Request for change process will be performed at level PT-3. Procedure is self-explanatory from CA-1 to CA-14. Taking decision on change is a multi-process, which means it may require different activities and artefacts. All documents impacted by change must also be updated when a change is implemented.

Responsibility: Manager/ Project Manager, Product Owner, Analysis team, CCB

B. Technical Aspects and Functionalities:

Roles and Responsibilities: The current roles and responsibilities related to Release Management processes include the following:

1. Preparation Phase

Each new planned component is generated via multiple means i.e. known as "preparation phase" as a whole. Each step is called "preparation activity" and numbered as PA-1, PA-2 etc. When a change is planned, this Release management process is started. Each release item should be assigned a release number. A new release item is created in PA-1, with all necessary inputs required in PA-2. All these related data are stored in one repository to be traced in future. To make the data entry process convenient, a User interface developed in any platform compatible with release repository will be helpful. When all those release items are created with required inputs that are inter related and inter dependent, then save all of them in form of a bundle (PA-3) to be approved by governance in implementation phase.

2. Governance

Governance body is responsible for release business requirements and for assembling final release bundles in consultation with Release Team Leadership. Their core activities include:

- Decision of the approval or rejection of received release bundle (A1)
- If its approved, then move forward to release engineer with a system notification (A1)
- This notification could be generated when whole project will be managed at a virtual platform like Microsoft Project, Primavera or e-GSN tool.
- At the completion of release tasks, to check whether it fulfils the "completed" status. (A16)
- After getting the notification of "completed" status, verify if it's closed (A17), else communicate with release engineer to take appropriate action (A18).

3. Release Team Leadership

Release Team Leadership directs to a group of technical specialists responsible for implementing a Release Bundle into deployable independent product. The Leadership Team is designed including Product Managers, Technical experts and Service Owners based on the types of releases. These established teams work with Governance for requirements prioritization considering the technical constraints in line with Release scope.

4. Release Engineer

The Release Engineer has decisive role to ensure the Releases are deployed according to processes and practices defined in the model. The Release Engineer plays expert role having all skills essential to understand all possible complications in Release execution. Their responsibilities include:

- Communicating with Governance, Release leadership Team, Change Advisory board, Clients, and QEC team.
- Documenting Release Bundle in the tool (MS Project, e-GSN etc).
- Creating placeholder Request for change (RFC) for the release based on available details and approved by Change Advisory board.
- Validating if all Release Tasks are deployed complete as per scope and requirements.
- Performing causality analysis and providing a solution support with the collaboration of release

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

team leadership.
- Communication of task status to the Release leadership team.

The activities are:
- Receive the approved release bundle
- Working with technical team members to start all release items independently from release bundle in any distributed team project tool (A2)
- List down the set of release tasks needs to be implemented (A2)
- Set the Timeline, budget, functional and non-functional requirements etc for each release task
- Create the release task in that project tool or system (A3)
- As per planned tasks and resources, assign them to suitable release team members appropriately (A4)
- After task assignment, a system notification will be generated to all concerned personnel. (A5)
- For each created task, decide where it needs to be passed for appropriate solution; to Release team members for development (Dev); to Quality Assurance (QA) team for performing required testing at this stage; or to Change Advisory Board for approving this new task as a change to be implemented.
- Few parallel tasks also execute when needed, like if a task is considered as change, it will be inserted in Change Placeholder (PT-1). This placeholder will trigger another parallel activity of Pre. Request for change process (PT-2) on the basis of requested change at (A3).

5. Release Team (directed by Release Engineer)

Release Teams refer to a group of team members responsible for the implementation of Release Bundle while committing to product scope. They work for Release Team Leadership on defined Release Tasks List and work assignments. Their progress is also monitored and evaluated in line with shared report/work's status. Their tasks are:
- To Receive work notifications on system with the whole predecessor task status (A6).
- Studying pre tasks status to check if its completed and allowed to resume further or wait for time being or communicate with release engineer about due date (A7).
- Developing the list of Release Tasks as per timeline and dependencies assigned and finely grained by Release Team Leadership. The Release Task List should include back out and handover to operations as mandatory steps. (A8)
- Prior to Execution of Release Tasks, update work status and documentation in tool (A9).
- On the basis of updated status, at A2, Release engineer will take further decision.

Other Responsibilities include
- Updating tool versions to avoid or dealing with expected errors in implementation process.
- Getting defects fixed from Quality assurance testing.
- Deploying Release Bundles in production environment after testing.

6. Quality Assurance Team

Quality Assurance (QA) Teams provided the Quality testing interfaces are accountable for release verification that the software releases meet their product scope and technical requirements. QA team will execute formal test artefacts and share testing results with Release leadership team to provide a complete release audit trail. This work/test status will help Release Team to get defects resolved by developers. Their tasks include:
- Studying the received assignment (A10)
- Developing test plans and cases. (A11)
- Executing test artefacts and update results. (A11)
- To help decision meeting, test results are shared through work status.(A12)
- Generate system notification for all concerned persons. (A13)
- Update defects and Move to A2, where Release Team resolves them.

7. Change Advisory Board

Change Advisory Board (CAB) represents the Change Management interfaces in line with Release Management. Their tasks include:
- Reviewing Request for changes and approving or rejecting them coordinating with Release Engineer. (A14)
- Communicate the decision (A15)
- A parallel activity will be started to perform the taken decision in Post Request for change (post RFC) process. (PT-3)

C. Key aspects of Release:

Releases are categorized in three major groups for work management at different levels i.e.
- Release bundles capture overall details of each and every major change in form of release.
- Release Items consist of all independent chunks of task those would be delivered as individual Releases.
- Release Tasks detail the work stipulated to execute all Release Items.
- Software Release

Software Release denotes a planned release for a software product. The Release contents consist of:

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

- Release Name and Description
- Service Components & Configuration Items
- Analyst or Release Engineer
- Planned or Actual start Dates

When a Release Engineer mark all releases as 100% complete and update the status of completion at platform only then a release would be deployed to customer. If there is a situation when any Release Item have to be eliminated from the Release Bundle then decision would be taken by analyst of release engineer to extract that item form current release bundle and may add into another created release bundle or discarded and declare the current release status as complete.

1. Release Items

Each Release consists of one or multiple Release Items. The items are actually the new major change occurred in previous project or essential modified feature that need to be deployed to Project Manager collectively as Release Bundle. The release bundle is made up of following features:

- Release Type like Defect/Problem or Service Query.
- List of functionalities needed to be deployed.
- Description of Request for Change linked to a specific Release Item.

2. Release Tasks

Release Tasks represent a set of actions required to be implemented in each Release. Release Tasks may include such information:

- Task type.
- The brief and long description.
- The assignments by specific group or an individual.
- Submission date.
- Work status and Percentage complete.
- Work notes

III RESULTS

As described in previous sections, a case study is performed to test this system on the overall development procedures of the product under study: “study on cloud”. It implemented a concrete architecture of a decision support system for software release management; and assessed the impact of the solution system on management tasks. Quality, progress and health are the major concerns in the present era to evaluate the success or failure of the deployed release (say, better quality, progress and health, means better release productivity). Now providing the prioritization and configuration of these three major concerns in the development and release management of “study on cloud”, leadership team is expected to choose precise solution in problem situation that will help them deploy the best release product. Following this approach, this research claims to provide the greater no. of successful Releases and in return, reducing the no. of Releases with unexpected results caused by inappropriate managerial solution support.

While performing the case study, the Release processes of “study on cloud” were automated and the effectiveness of decision support was tested for “online examination system”, “faculty portal” and “plagiarism checker” that on which grounds problem oriented decisions were provided. The overall results of this approach are as follows:

Planning: In the automated development of this web application releases, to support problem oriented solution support scenario, management constructed a hierarchical definition for each of quality, progress, and health concerns for each new release component. The initial and expected final values for each of these concerns are computed. For each of the three dimensions, an ideal line is drawn connecting the initial and expected final values. These lines represent the initial plan of the project. When there was a need for plan modification; team leaderships were used for appropriate adjustments. Due to availability of DSSRM, it improved the coordination and communication between leadership team and stake holders to ensure smooth and parallel managerial and development process on time. All procedures and computed result updates were saved in single common repository and maintained on one combined document. This way the original plan remains intact for possible future improvement of the planning process.

Operational: At regular intervals quality, health, and progress are evaluated according to their definitions. The computed value at a specified interval is compared against the expected value; a point on the plan line. When the computed value fell below the expected value, it indicated a possible problem. To identify possible causes to a potential problem, tactical analysis was initiated.

Tactical: Upon identification of a problem, management began searching for correlations and possible causes for the problem. As the problem was caused by a drop in the quality, by drilling into the numbers, management identified goals that failed to contribute to the value of the overall hierarchy. Based on the identification of the goals that have not been accomplished; other dimensions investigated to determine if additional aspects of the project such as health and progress had any impact on quality. When the problem seemed to be the result of aggressive planning, the overall plan adjusted according to the prediction at the current point in the same single document. The release problem seemed to endure for the remainder of the project and possible future releases of the product, a strategic analysis performed to find potential solutions and supporting evidence for future directions.

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Strategic: When management identified potential future problem or re-occurrence of a similar problem, a longer term precise solution defined. The solution was the result of the input from the strategic layer, from release team leadership or internal to the team. In this case, a hypothesis is formulated that it improved the productivity of releases successfully and its validity is studied based on the results of DSSRM. Should the hypothesis prove to be of value; an impact analysis is performed after establishing the proposed release processes; a strategy for the implementation of the solution including its evaluation is constructed. This case study used some forms of defect analysis for their strategic analysis. As a result, a set of queries were formulated that, decision support causality analysis performed various analysis on defect arrivals, defect turnaround time, code growth, and customer reported problems.

Prior to results of decision support causality analysis in "study on cloud" release management, Figures 4, 5, and 6 show the computed values for progress, health, and quality goals respectively. These values were computed at an interval during the case study. Due to confidentiality issues, the times are not shown on the charts. These charts present the deviations from the plan that was used to initiate tactical analysis.

A. Quality

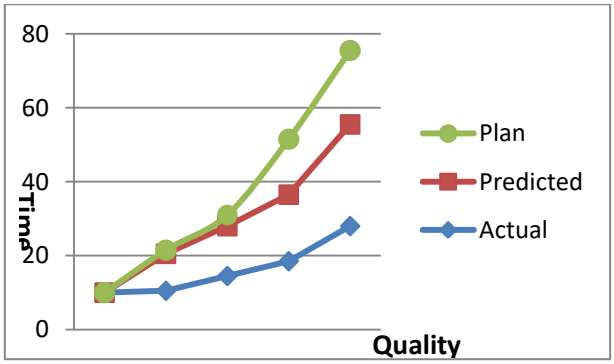

Fig 1 Quality Result, Quality Analysis of the Project

This figure shows planned, computed, and predicted values of the quality goals of all components of the project "study on cloud". As the figure shows, the quality goals at the start of the project were behind the plan. As the time passed, the quality goals moved closer to their plan. The prediction at the last sample point indicates small future deviation. At this point, the team can either adjust their quality plans or shift resources to improve quality.

B. Progress

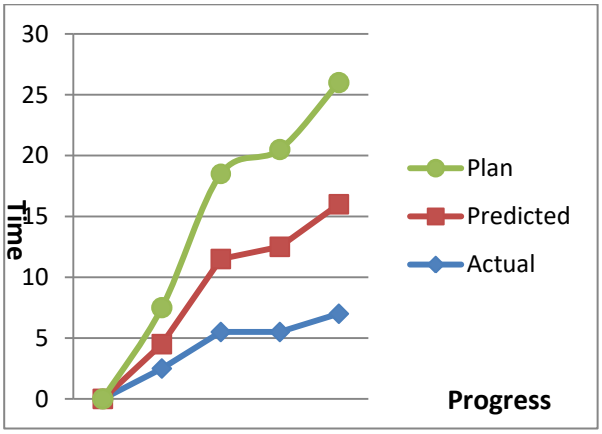

Fig 5 Progress Result, progress analysis of the overall project

This figure shows planned, computed, and predicted values of the progress goals of project "study on cloud". As the figure shows, at the early stages, the progress goal significantly deviated from the plan but later on it recovered. However, the prediction of the last computed point indicates a large deviation from the final release plan. Such a figure can indicate aggressive planning or poor resource allocation.

C. Health

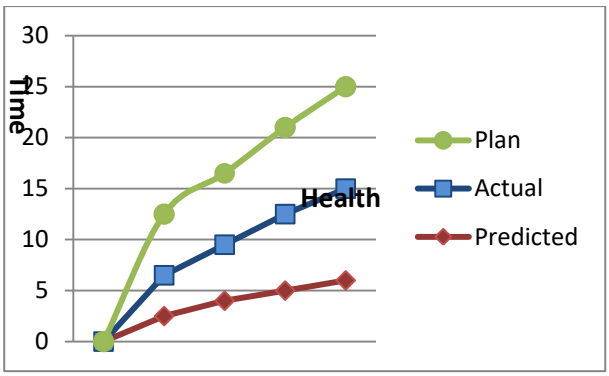

Fig 6 Health Result, Health analysis of the project

This figure shows planned, computed, and predicted values of the health goals of the project "study on cloud". As the figure shows, the health of the system initially was close to the plan. However, as time passes, the health deviates more and more from the plan. This deviation can be explained in terms of pressures on the progress goal or a change of concern in terms of health goals. The subjectiveness of health definition requires a replay of the health goal based on other definitions of health.

IV DISCUSSION

The case study presented in previous chapters aimed not to interfere with the progression of the "study on cloud" product

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

development and its management. However, early on, the management showed an interest in the gradual applications of the ideas presented in this research. As a result, the proposed management framework became integrated into the overall management strategy. A key argument presented by the management suggested that to effectively assess the potential improvements, the proposed framework has to be gradually integrated [21] and its effects should remain continual and small [22]. Upon discussing the ideas of this study, the management formulated their requests as follows:

- identification of major problems;
- determination of the significance of the problems;
- determination of possible causes;
- evaluation of potential solutions; and
- Assessment of future impact.

The prototype software release management provided first line managers with regular and timely reports on features and defects, their status, and overall impact. Each manager additionally provided more detailed requirements for presentation of their reports on “online examination system”, “faculty portal” and “plagiarism checker”.

A commercial continuously growing web application development has numerous diverse approaches enacted [23]. In previous sections, this study focused on viewing evolutionary software issues as a matter of concern for management especially in decision making. Previously, such issues were discarded, hidden in log files, or at best poorly structured and maintained. During the case study, as we leveraged the available information to provide reports on development processes and to identify possible trends, we observed that more decision related managerial dodges were identified for integrating new features into existing product for providing potential solution in a situation.

In other words, the direction of the maturity of management processes was from having little or no existing processes towards more new processes in more structured way [24]. The number of processes increased drastically over the period of the study with the growth of product (almost doubled), which in turn exacerbated the need for automation of release system processes and concerned solution support. As more analysis was performed, process flow became more complex. Upon discovery of the faults in the analysis of the scenario “study on cloud” development and release management, few defects are opened and discussed previously.

To summarize, the creation of the decision support system for software release management for the case study has resulted in more questions being asked and novel answers being offered than previous releases. The availability of all processes to managers has resulted in more focused planning, better tactical analysis and improved strategy formulation. The solution support in software release management can be summarized as follows:

- Operational analysis:
  - Improved planning, monitoring, and projection processes.
  - Improved communication among various stakeholders.
  - Timely identification of problems and their locality.
- Tactical analysis:
  - Improved causal analysis of identified problems.
  - Access to subject-oriented, integrated, and temporal data for the project.
  - Enabled benchmarking of various tasks for future releases.
- Strategic analysis:
  - Facilitated long term planning and simulation.
  - Enabled assessment of analysis of potential solutions.
  - Provided a means of improving planning processes and in result the improved productivity of releases.

Upon detection of a potential problem, higher-level managers were informed, and evidence data, possible causes, potential solutions, and future impacts were presented to them. The following subsections present various benefits and limitations of the release management support system. It must be noted that some problems faced by the management are not noticeable at a smaller scale.

V CONCLUSION

This study claimed that viewing software as statistics and leveraging business intelligence methods and technologies, in particular release management, will improve the management of software product development. The underlying pillars of this claim were: software evolution, multiplicity of concern, continual improvement and especially decision support. An argument for Decision making, vital to Software Release Management, is presented and includes deciding on requirements prioritization and the contents of coming releases while the phases of software development life cycle reviewed in which the release management is employed. A means for defining and capturing multiplicity of concerns proposed; and also outlined architecture for management support. The case study demonstrated an application of the concepts presented in earlier sections while emphasizing the importance of decision support on continuous improvement in lieu of software evolution. This section concludes the research by summarizing its key contributions and outlining possible future directions. The following list summarizes the contributions of this study.

A decision supported release management framework:

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

- Focused management goals and their evaluation on the local projects;
- Provided a means for defining measurable goals facilitating monitoring and predicting their values;
- Provided a means of organizing defined goals along various dimensions of interest; and
- Provided flexibility for changing and modifying locally defined goals in lieu of unforeseen changes.

A decision support architecture:

- Focused on managing by following defined procedures;
- Developed an architecture for process flow and maintenance of integrated, temporal, and subject-oriented software releases;
- Implemented a management support for multiplicity of concerns;
- Provided a means of storing project histories for replay; and
- Provided flexibility for the evolution of software releases based on defined work flow.

A case study:

- Demonstrated an application of the proposed ideas to an industrial case study. This dissertation showed that the ideas are applicable and scale to handle large projects.
- Implemented a concrete architecture of a decision support system for software release management; and
- Assessed the impact of the solution system on management tasks.

REFERENCES

[1] Ferreira N. N. V. and Langerman, J. J., Proving that the release management process can enhance throughput in software development projects, Computer Science & Education (ICCSE), 2014 9th International Conference on, Vancouver, BC, pp. 419-424, 2014.

[2] Banta, V. C. and Cojocaru, D., Development Center Tool a software application for change request management, System Theory, Control and Computing (ICSTCC), 2013 17th International Conference, Sinaia , pp. 42-47, 2013.

[3] Regnell, B. and Kuchcinski, K., Exploring Software Product Management decision problems with constraint solving - opportunities for prioritization and release planning, Software Product Management (IWSPM), 2011 Fifth International Workshop on, Trento, pp. 47-56, 2011.

[4] Ngo-The, A. and Ruhe, G., Optimized resource allocation for software release planning, IEEE Transactions on Software Engineering, vol. 35, no. 1, pp. 109 – 23, 2009.

[5] Solyman, A. M, Ibrahim, O. A. and Elhag, A. A. M., Project management and software quality control method for small and medium enterprise, Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE), 2015 International Conference on, Khartoum, pp. 123-128, 2015.

[6] Yue, H. Liu, X. and Zhao, S., Evaluate Two Software Configuration Management Tools: MS Perforce and Subversion, Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on, Wuhan, pp. 1-6, 2010.

[7] Salfischberger, T., van de Weerd, I. and Brinkkemper, S., The Functional Architecture Framework for organizing high volume requirements management, Software Product Management (IWSPM), 2011 Fifth International Workshop on, Trento, pp. 17-25, 2011.

[8] Geogy, Manju, and Andhe-Dharani. Prominence of each phase in Software development life cycle contributes to the overall quality of a product. Soft-Computing and Networks Security (ICSNS), 2015 International Conference on. IEEE, 2015.

[9] Garrido, P. J., Vizcaíno, A., Andrada, J., Monasor, M. J. and Piattini, M., DPMTool: A Tool for Decisions Management in Distributed Software Projects, Global Software Engineering Workshops (ICGSEW), 2012 IEEE Seventh International Conference on, Porto Alegre, pp. 22-27, 2012.

[10] Li, C., van den Akker, M., Brinkkemper, S. and Diepen, G.,An integrated approach for requirement selection and scheduling in software release planning, Requirements Engineering, vol. 15, pp. 375 396, 2010.

[11] Michlmayr, M., Fitzgerald, B. and Stol, K. J., Why and How Should Open Source Projects Adopt Time-Based Releases?, in IEEE Software, vol. 32, no. 2, pp. 55-63, Mar.-Apr. 2015.

[12] Al-Zawahreh, Hanan, and Almakadmeh, Khaled., Procedural Model of Requirements Elicitation Techniques. Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication. ACM, 2015.

[13] Heng-You Lin, Sierla, S., Papakonstantinou, N. and ValeriyVyatkin, A change request management in model driven engineering, Automation Science and Engineering (CASE) 2015 IEEE International Conference on, pp. 1054-1059, 2015.

[14] Ren, Y., Quan, Q., Xing, T. and Chen, X., Fuzzy Decision Analysis of the Software Configuration Management Tools Selection, Information Science and Engineering (ISISE), 2010 International Symposium on, Shanghai, pp. 295-297, 2010.

[15] Shah, Unnati S. An Excursion to Software Development Life Cycle Models: An Old to Ever-growing Models. ACM SIGSOFT Software Engineering Notes 41.1): 1-6 (2016)

[16] Alawairdhi, Mohammed. Agile development as a change management approach in software projects: Applied case study. 2016 2nd International Conference on Information Management (ICIM). IEEE, 2016.

[17] Giesecke, S. Friebe, J. and Frenzel, M., Long-Term Software Architecture Management with Multi-technology Tool Support, Software Maintenance and Reengineering (CSMR), 2011 15th European Conference on, Oldenburg, pp. 321-324, 2011.

[18] Parveen, K. and Munir, F., Visual backlog in agile management tools for rapid software development, 2015 International Conference on Open Source Systems & Technologies (ICOSST), Lahore, pp. 84-90, 2015.

[19] Seo, D. Shin, D. and Bae, D. H., Quality Based Software Project Staffing and Scheduling with Cost Bound, 2015 Asia-Pacific Software Engineering Conference (APSEC), New Delhi, pp. 269-276, 2015.

[20] GarmLucassen, Jan Martijn E. M. van der Werf and SjaakBrinkkemper, Alignment of software product management and software architecture with discussion models, Software Product Management (IWSPM) 2014 IEEE IWSPM 8th International Workshop on, pp. 21-30, 2014.

[21] Najjar, Sireen Kamal, and Khalid T. Al-Sarayreh. Capability Maturity Model of Software Requirements Process and Integration (SRPCMMI). Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication. ACM, 2015.

[22] Ya-hong, L., Jian, L. and Ke-gang, H., The software project progress measurement frame based on GQM model, Management Science and Engineering (ICMSE), 2013 International Conference on, Harbin, pp. 133-138, 2013.

[23] Seele, W., Syed S. and SjaakBrinkkemper, The Functional Architecture Modeling Method Applied on Web Browsers, Software Architecture (WICSA) 2014 IEEE/IFIP Conference on, pp. 171-174, 2014.

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

[24] Ferreira, Natasha N. Vitó, and Josef J. Langerman. Proving that the release management process can enhance throughput in software development projects. Computer Science & Education (ICCSE), 2014 9th International Conference on. IEEE, 2014.

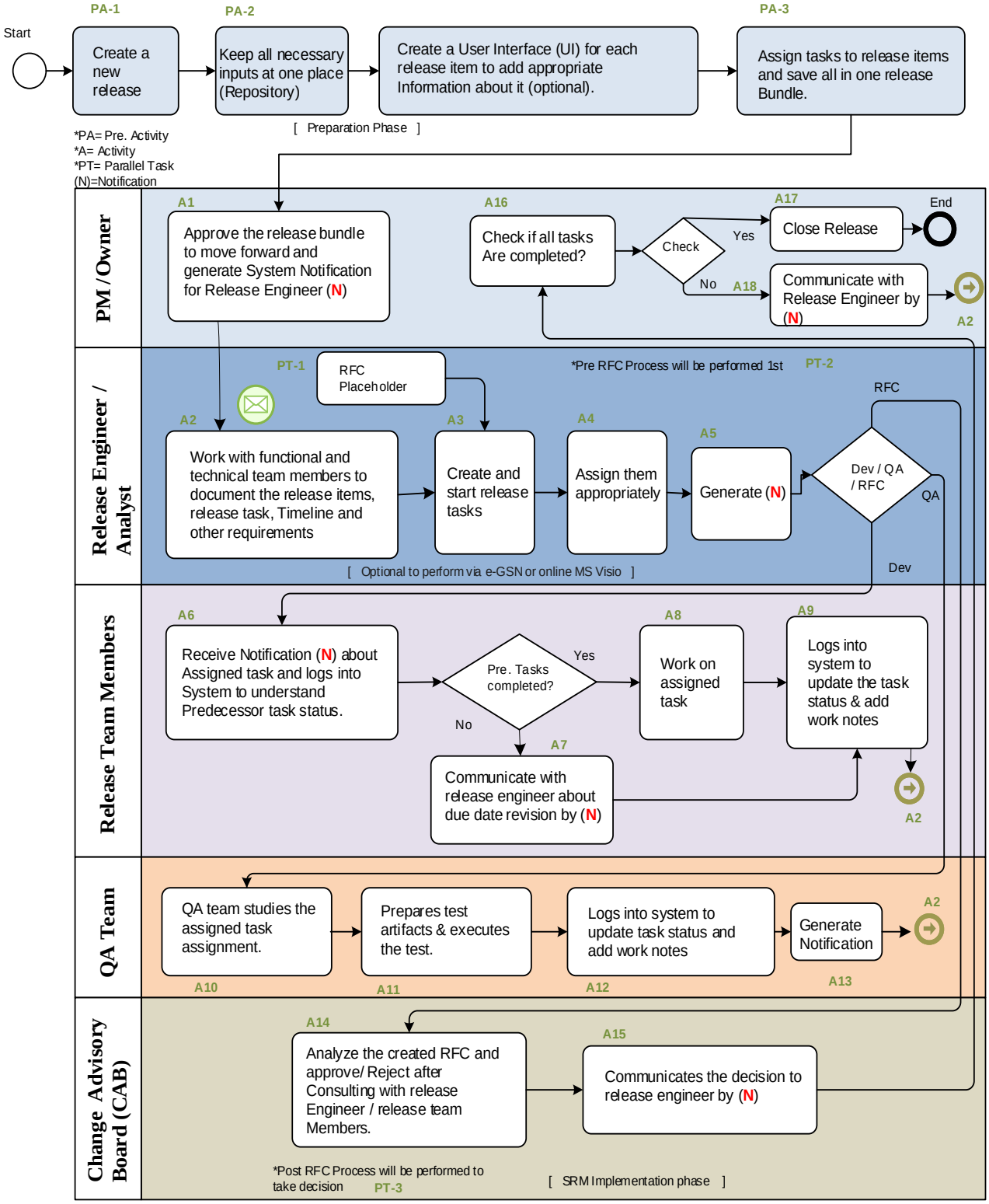

Fig 2: DSSRM, a Decision Support System for Software Release Management in Software Development Life Cycle

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

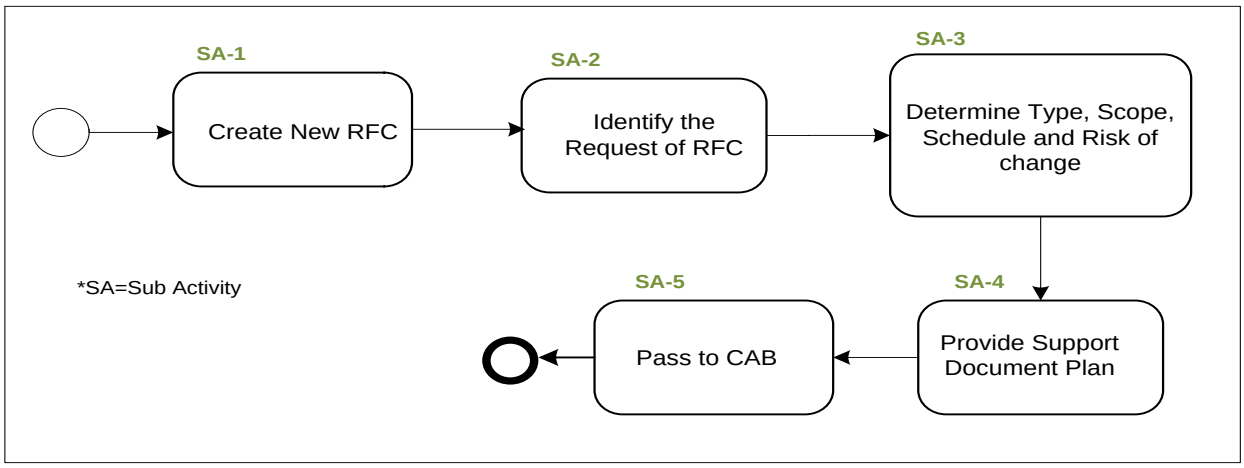

Fig 3: pre. RFC, pre Request for change (RFC) process before passing change to Change advisory board

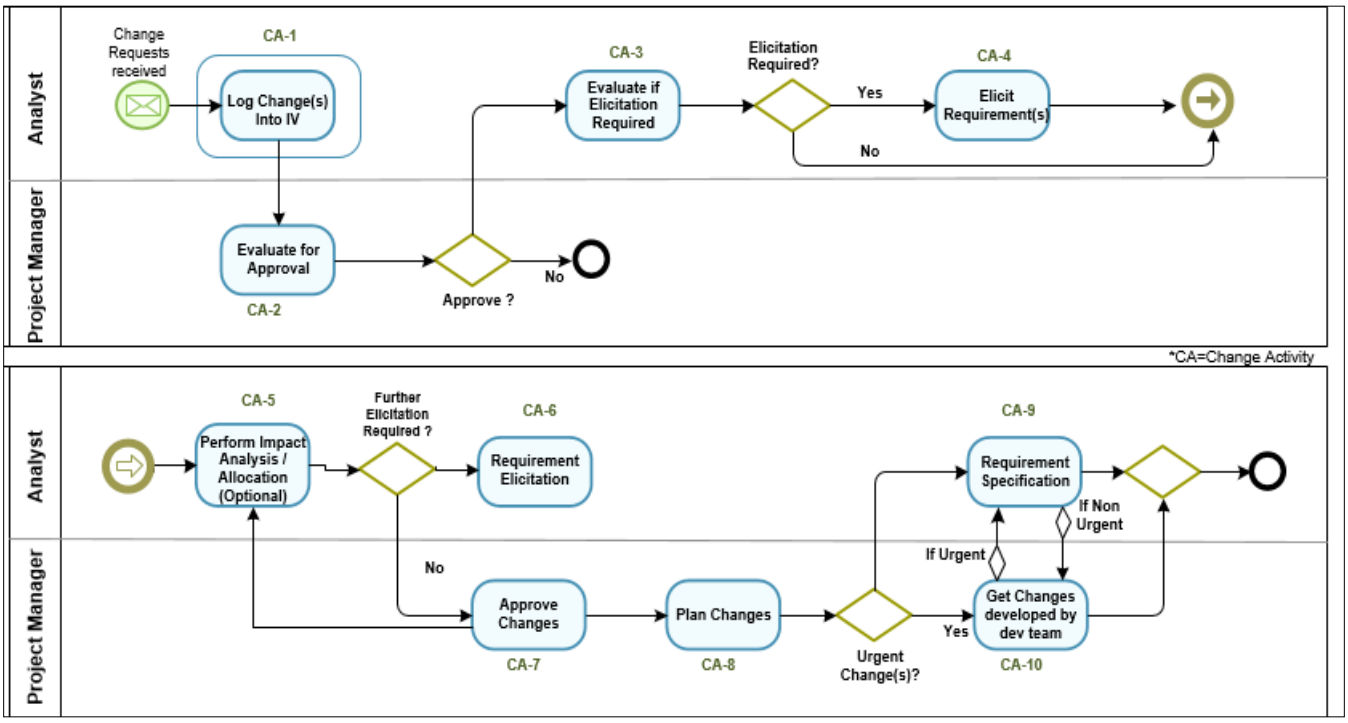

Fig 4: Post RFC, a request passed to Change Advisory Board

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)